A Branch-and-Price-and-Cut Algorithm for Discrete Network Design Problems Under Traffic Equilibrium

David Rey[1] and Michael W. Levin[2]

[1]SKEMA Business School, Université Côte d'Azur, Sophia Antipolis, France, `david.rey@skema.edu`

[2]Department of Civil, Environmental, and Geo- Engineering, University of Minnesota, Minneapolis, USA, `mlevin@umn.edu`

Abstract

This study addresses discrete network design problems under traffic equilibrium conditions or DNDPs. Given a network and a budget, DNDPs aim to model all-or-nothing decisions such as link addition to minimize network congestion effects. Congestion is measured using traffic equilibrium theory where link travel times are modeled as convex flow-dependent functions and where users make selfish routing decisions. In this context, the collective route choice of users is a Wardropian equilibrium and DNDPs admit a bilevel optimization formulation where the leader represents the network designer, and the follower is a parameterized traffic assignment problem (TAP). This study introduces a novel and exact branch-and-price-and-cut (BPC) algorithm for DNDPs that exploits the structure of the problem and harnesses the potential of path-based formulations for column generation (CG). Leveraging the convexity and the separability of the objective function, we develop successive relaxations of the bilevel problem that lead to an efficient outer approximation scheme that relies on solving a sequence of linear programs. We combine this OA procedure with a CG approach whose pricing subproblem can be solved in polynomial-time. This scheme is embedded within a single tree BPC algorithm to determine lower bounds while upper bounds are computed by solving parameterized TAPs. Numerical experiments conducted on a range of DNDP instances based on three transportation networks reveal that our BPC algorithm significantly outperforms state-of-the-art methods for DNDPs. Notably, we close several open instances of the literature, and we show that our BPC algorithm can solve DNDP instances based on networks whose number of nodes and of commodities is one order of magnitude larger than any previously solved instance.

Keywords: Bilevel optimization, Branch-and-price-and-cut, Network design, Traffic equilibrium, Selfish routing

1 Introduction

We consider discrete network design problems under traffic equilibrium. Discrete network design problems aim to model all-or-nothing decision problems, such as link or node addition, in a network. Traffic equilibrium refers to selfish routing and congestion effects in a network. In discrete network design problems under traffic equilibrium or DNDPs for short, the goal is to minimize congestion effects while accounting for users' collective route choice in response to the design. Such problems have well-documented applications in transportation and routing (Magnanti and Wong 1984) and can be used to inform decision-making in problems where the collective behavior of users influences congestion and therefore the choice of the network design. It-is well-known that DNDPs are NP-hard even if link travel time functions—also known as latency functions or delay functions—are linear (Roughgarden 2006). In transportation planning applications, multiple commodities representing the travel of users between their origin and destination often coexist and congestion functions are typically modeled as convex nonlinear functions of link flow (Colson et al. 2007). DNDPs are often modeled using bilevel optimization formulations where the leader represents the network designer and the follower represents a parameterized traffic equilibrium problem (Yang and H. Bell 1998). In this context, identifying the network design that optimizes the designer's objective function is notoriously challenging. Existing methods to solve DNDPs exactly do not scale well, mainly due to their inability to optimize over large networks. This difficulty stems from the nonlinearity of link travel time functions but also from the necessity to compute network equilibrium flows repeatedly to find optimal solutions.

The contributions of this study are as follows: we develop a novel methodology to solve DNDPs to optimality at scale. Unlike most existing approaches to solve DNDPs, we formulate a path-based convex relaxation of the original bilevel optimization problem. We combine outer approximation (OA) (Duran and Grossmann 1986, Fletcher and Leyffer 1994) and column generation (CG) (Dantzig and Wolfe 1960, Lübbecke and Desrosiers 2005) to obtain a tractable linear model that efficiently computes lower bounds. We propose a single tree branch-and-cut-and-price (BPC) algorithm that gradually adds cuts and path variables to this linear model and exploits the performance of state-of-the-art traffic assignment algorithms to computer upper bounds. We enrich our BPC algorithm with initialization heuristics and new value function cuts. Extensive numerical experiments that compare the performance of our BPC algorithm with three benchmarks, including two exact algorithms from the literature, highlight the benefits of the proposed approach. Notably, these experiments show that the solving the linear path-based model via a CG approach is computationally efficient and avoids scaling issues from solving a link-based multicommodity network design problem. Our numerical results solve a number of unsolved DNDP instances and we introduce new larger problem instances on medium to large-sized transportation networks to the community. For reproducibility purposes, all data and codes of this study are made publicly available.

2 Methodology

DNDPs can be formulated as bilevel optimization problems where the leader problem aims to identify the optimal selection of a discrete resource to add to a network to minimize the total system travel time (TSTT) and the follower problem represents users' reaction, typically as a static traffic assignment problem (TAP) under Wardrop's user equilibrium (UE) principle (Wardrop 1952). For presentation and experimentation purposes, we use the link addition DNDP—hereby referred to as DNDP—as our target discrete network design problem under traffic equilibrium since this is the most studied DNDP in the literature.

The DNDP can be defined on a network with a node set $\mathcal{N}$ and link set $\mathcal{A}$ as a multicommodity network flow problem with nonlinear link travel time functions. Let d_w be the demand for commodity

$w \in \mathcal{W} \subset \mathcal{N} \times \mathcal{N}$. Let Π_w be the set of paths connecting commodity $w \in \mathcal{W}$ and let $\Pi = \cup_{w \in W} \Pi_w$ be the set of all paths. Let $[\delta_a^\pi]_{a \in \mathcal{A}, \pi \in \Pi}$ be the link-path incidence matrix of the network. We denote h_π the flow on path $\pi \in \Pi$ and $\boldsymbol{h} = [h_\pi]_{\pi \in \Pi}$ the vector of path flows. Let $t_a(\cdot)$ represent the travel time function on link $a \in \mathcal{A}$, typically modeled as a positive and increasing function of the total link flow x_a to ensure the uniqueness of the UE link flows. Let $\mathcal{A}_1$ be the set of existing links and let $\mathcal{A}_2$ be the set of candidate links to improve the network, $\mathcal{A} = \mathcal{A}_1 \cup \mathcal{A}_2$. For each link $a \in \mathcal{A}_2$, let g_a be the cost of adding this link to the network and let $y_a \in \{0, 1\}$ be the variable representing this choice. Let B be the available budget for optimization.

The leader represents a network designer that aims to minimize the TSTT defined as the sum of $x_a t_a(x_a)$ over all links $a \in \mathcal{A}$, subject to a budget constraint capturing the cost of link addition decisions $\boldsymbol{y}$, hereby referred to as leader variables. The link flow pattern variables $\boldsymbol{x} = [x_a]_{a \in \mathcal{A}}$ are determined by the follower problem, which is the TAP formulation under UE (Beckmann et al. 1956, Leblanc 1975, Magnanti and Wong 1984). The impact of the leader variables $\boldsymbol{y}$ in the follower is achieved through the linking indicator constraints that require null flow on $\mathcal{A}_2$ links that remain closed.

The follower problem is a TAP based on Beckmann et al. (1956)'s formulation parameterized by leader variables $\boldsymbol{y}$:

$$\begin{aligned}
\texttt{TAP}(\boldsymbol{y}) : \min_{\boldsymbol{x}, \boldsymbol{h}} \quad & \sum_{a \in \mathcal{A}} \int_0^{x_a} t_a(v) dv & & \text{(1a)} \\
\text{s.t.} \quad & \sum_{\pi \in \Pi_w} h_\pi = d_w & \forall w \in \mathcal{W} \quad & \text{(1b)} \\
& \sum_{\pi \in \Pi} h_\pi \delta_a^\pi = x_a & \forall a \in \mathcal{A} \quad & \text{(1c)} \\
& x_a = 0 \qquad \text{if } y_a = 0 & \forall a \in \mathcal{A}_2 \quad & \text{(1d)} \\
& h_\pi \geq 0 & \forall \pi \in \Pi \quad & \text{(1e)}
\end{aligned}$$

Let $\texttt{TAP}(\boldsymbol{y})$ also denote the set of optimal link flow solutions $\boldsymbol{x}$ of the parameterized TAP (1). The DNDP can be formulated as the following bilevel optimization problem:

$$\begin{aligned}
\min_{\boldsymbol{y}} \quad & \sum_{a \in \mathcal{A}} x_a t_a(x_a) & & \text{(2a)} \\
\text{s.t.} \quad & \sum_{a \in \mathcal{A}_2} y_a g_a \leq B & & \text{(2b)} \\
& y_a \in \{0, 1\} & \forall a \in \mathcal{A}_2 \quad & \text{(2c)} \\
& \boldsymbol{x} \in \texttt{TAP}(\boldsymbol{y}) & & \text{(2d)}
\end{aligned}$$

Note that although the follower problem contains two groups of variables: path flows and link flows, only the latter are used to compute the leader objective function. Furthermore, it is well-known that, if link travel time functions $t_a(\cdot)$ are positive and increasing then, for any leader decision $\boldsymbol{y}$, the objective function of the follower is strictly convex with regards to variables $\boldsymbol{x}$ and there is a unique UE link flow pattern $\boldsymbol{x} \in \texttt{TAP}(\boldsymbol{y})$ (Sheffi 1985).

We exploit the properties of the objective function of DNDP and propose a novel link-based Outer Approximation (OA) model to obtain a Mixed-Integer Linear Programming (MILP) relaxation of the SO relaxation. Leveraging the path-based formulation of this MILP relaxation, we develop a Column Generation (CG) with a polynomial-time pricing subproblem.

For presentation purposes, we define the following sets: let $\mathcal{X}$ be the set of (unrestricted) feasible link flows:

$$\mathcal{X} \equiv \left\{ \boldsymbol{x} \in \mathbb{R}^{|\mathcal{A}|} : \sum_{\pi \in \Pi} h_\pi \delta_a^\pi = x_a, \ \forall a \in \mathcal{A}, \sum_{\pi \in \Pi_w} h_\pi = d_w, \ \forall w \in W, h_\pi \geq 0, \ \forall \pi \in \Pi \right\} \tag{3}$$

Let $\mathcal{Y}$ be the set of feasible link addition decisions:

$$\mathcal{Y} \equiv \left\{ \boldsymbol{y} \in \{0,1\}^{|\mathcal{A}_2|} : \sum_{a \in \mathcal{A}_2} y_a g_a \leq B \right\} \tag{4}$$

2.1 System Optimum Relaxation

Let $Q = \sum_{w \in \mathcal{W}} d_w$ be the total demand. A mathematical programming formulation of the system optimum DNDP, denoted SO-DNDP is:

$$\min_{\boldsymbol{y} \in \mathcal{Y}, \boldsymbol{x} \in \mathcal{X}} \quad \sum_{a \in \mathcal{A}} x_a t_a(x_a) \tag{5a}$$

$$\text{s.t.} \quad x_a \leq y_a Q \qquad \forall a \in \mathcal{A}_2 \tag{5b}$$

Here the linking constraint (1d) is reformulated into a so-called big-M constraint (5b) which is linear and for which a finite bound is known. The SO relaxation (5) of the DNDP formulation (2) is a MINLP with a convex objective function, which motivates the use of dedicated algorithms, notably OA methods.

2.2 Outer Approximation

The nonlinear link travel time functions $t_a(\cdot)$ make problem (5) difficult to solve. It is well-known that the objective function (2a) of the DNDP (and of SO-DNDP) is convex, which we use to write an OA of the objective function (Duran and Grossmann 1986, Fletcher and Leyffer 1994). Furthermore, the objective function of DNDP (TSTT) is separable with regards to link flow variables $\boldsymbol{x}$ (Beckmann et al. 1956), which means that the OA can be made tighter by creating a link-based OA instead of a single OA of the entire objective function. We exploit these properties to build a link-based OA of the terms $x_a t_a(x_a)$ composing the objective function.

Formally, given a vector $\boldsymbol{x}^k \in \mathcal{X}$ of feasible link flows, the gradient of $x_a t_a(x_a)$ at $\boldsymbol{x}^k$ is:

$$x_a^k t_a(x_a^k) + \frac{d\left[x_a^k t_a(x_a^k)\right]}{dx_a^k} \times (x_a - x_a^k) = x_a^k t_a(x_a^k) + (x_a^k t_a'(x_a^k) + t_a(x_a^k)) \times (x_a - x_a^k) \tag{6a}$$

$$= x_a \left(x_a^k t_a'(x_a^k) + t_a(x_a^k)\right) - (x_a^k)^2 t_a'(x_a^k) \tag{6b}$$

$$= x_a \alpha_a^k + \beta_a^k \tag{6c}$$

where $\alpha_a^k \equiv x_a^k t_a'(x_a^k) + t_a(x_a^k)$ and $\beta_a^k = -(x_a^k)^2 t_a'(x_a^k)$ are constants. Because of convexity of the TSTT objective function, for any $\boldsymbol{x}^k \in \mathcal{X}$, $x_a \alpha_a^k + \beta_a^k$ is a linear under-estimator of $x_a t_a(x_a)$.

Let $\mu_a \geq 0$ be a real decision variable representing the contribution of link $a \in \mathcal{A}$ to the objective function (2a). Let $\mathcal{C}_a$ be the set of indices k corresponding to the link flows x_a^k at which OA is performed, i.e. $\mathcal{C}_a$ represents the set of OA cuts used to under-estimate the contribution of link $a \in \mathcal{A}$ to the objective function. Given a collection of index sets $[\mathcal{C}_a]_{a\in\mathcal{A}}$, the following formulation is a MILP relaxation of SO-DNDP:

$$\begin{aligned}
&\min_{\boldsymbol{y}\in\mathcal{Y},\boldsymbol{x}\in\mathcal{X},\boldsymbol{\mu}\geq\boldsymbol{0}} && \sum_{a\in\mathcal{A}} \mu_a && && \text{(7a)}\\
&\text{s.t.} && \mu_a \geq x_a \alpha_a^k + \beta_a^k && \forall a \in \mathcal{A}, k \in \mathcal{C}_a && \text{(7b)}\\
& && x_a \leq y_a Q && \forall a \in \mathcal{A}_2 && \text{(7c)}\\
& && y_a \in \{0,1\} && \forall a \in \mathcal{A}_2 && \text{(7d)}
\end{aligned}$$

Solving Formulation (7) yields a lower bound (LB) on the optimal objective function value (OFV) of DNDP. In contrast to the DNDP literature, this formulation exploits the link-separability property of the objective function of the DNDP. Farvaresh and Sepehri (2013) developed a MILP relaxation of SO-DNDP based on the OA of the entire objective function as opposed to link-based OAs. Their MILP formulation was also formulated using a link-based multicommodity network flow model. Instead, Formulation (7) is path-based and contains an exponential number of path flow variables $\boldsymbol{h}$. We exploit the MILP structure of Formulation (7) and its amenability to CG techniques to avoid enumerating all paths within Formulation (7).

2.3 Column Generation

To solve Formulation (7), we consider its linear programming (LP) relaxation. Let $\bar{\Pi} \subset \Pi$ be a restricted set of paths and let $\bar{\Pi}_w \subset \Pi_w$ be the corresponding commodity-based restricted path sets. The restricted master problem (RMP) is:

$$\begin{aligned}
&\min_{\boldsymbol{y},\boldsymbol{x},\boldsymbol{h},\boldsymbol{\mu}} && \sum_{a\in\mathcal{A}} \mu_a && && \text{(8a)}\\
&\text{s.t.} && \mu_a \geq x_a \alpha_a^k + \beta_a^k && \forall a \in \mathcal{A}, k \in \mathcal{C}_a && \text{(8b)}\\
& && \sum_{a\in\mathcal{A}_2} y_a g_a \leq B && && \text{(8c)}\\
& && \sum_{\pi\in\bar{\Pi}_w} h_\pi = d_w && \forall w \in \mathcal{W} && \text{(8d)}\\
& && \sum_{\pi\in\bar{\Pi}} h_\pi \delta_a^\pi = x_a && \forall a \in \mathcal{A} && \text{(8e)}\\
& && x_a \leq y_a Q && \forall a \in \mathcal{A}_2 && \text{(8f)}\\
& && 0 \leq y_a \leq 1 && \forall a \in \mathcal{A}_2 && \text{(8g)}\\
& && h_\pi \geq 0 && \forall \pi \in \bar{\Pi} && \text{(8h)}
\end{aligned}$$

Since the objective of (SO-)DNDP is to minimize network-wide congestion, link and path flows are indirectly minimized. Hence, constraints (8d) and (8e) can be rewritten as inequalities to restrict the sign of their dual variables, namely:

$$\sum_{\pi \in \Pi_w} h_\pi \geq d_w \qquad \forall w \in W \tag{9a}$$

$$\sum_{\pi \in \Pi} h_\pi \delta_a^\pi \leq x_a \qquad \forall a \in A \tag{9b}$$

We denote $\sigma_w \geq 0$ the dual variable of the demand constraint (9a) and $\zeta_a \geq 0$ the dual variable of the link flow constraint (9b). Given a commodity $w \in \mathcal{W}$ and a path $\pi \in \Pi_w$, the reduced cost of variable h_π, denoted c_π, is:

$$c_\pi = -\sigma_w + \sum_{a \in A} \delta_a^\pi \zeta_a \tag{10}$$

The reduced costs of path-flow variables $\boldsymbol{h}$ can be computed in polynomial-time by solving, for each commodity $w \in \mathcal{W}$, a shortest path problem in the directed network $(\mathcal{N}, \mathcal{A})$ with link costs given by the dual vector $\boldsymbol{\zeta} = [\zeta_a]_{a \in \mathcal{A}}$ and deducing σ_w from the shortest path length.

This CG approach is novel and provides a paradigm shift for solving DNDPs at scale: as shown in our numerical experiments in Section 3, the pricing of path flows variables is computationally inexpensive and the path-based formulation of the SO-relaxed OA model of DNDP is more efficient than its link commodity-based counterpart. We integrate the OA approach and the CG within a single tree branch-and-price-and-cut (BPC) algorithm to solve DNDPs under traffic equilibrium.

3 Numerical Results

We conduct numerical experiments to test the performance of the BPC algorithm for the link addition DNDP. We consider three benchmarks: a branch-and-cut (BC) algorithm which follows the same single tree BB algorithm as BPC but uses a commodity link-based formulation instead of the path-based formulation, therefore no CG is required. Therefore, the only difference between BPC and BC is the use of a path-based formulation and the CG procedure. We also compare BPC and BC with two benchmark algorithms from the literature: Leblanc (1975)'s BB algorithm—hereby referred to as Leblanc—which solves SO-TAPs to determine LBs; and Farvaresh and Sepehri (2013)'s BB algorithm which uses Fletcher and Leyffer (1994)'s OA algorithm to determine LBs—hereby referred to as FS_NETS. All four implemented algorithms, i.e. BPC, BC, FS_NETS and Leblanc and; use the same `check`(k) procedure to scan a BB node k before further processing it if it is labeled `unfixed`. All TAPs are solved using our implementation of Bar-Gera (2010)'s TAPAS algorithm.

We use three transportation networks from a public repository containing traffic assignment, i.e. network and trips, data (Transportation Networks for Research Core Team 2024) to generate DNDP instances: SiouxFalls (SF), Eastern Massachusetts (EM) and BerlinMitteCenter (BMC). SF is a test network with 24 nodes, 76 links and 528 commodities widely used in DNDP studies (Farvaresh and Sepehri 2013, Wang et al. 2013, Rey 2020). EM contains 74 nodes, 258 links and 1113 commodities; and BMC contains 398, 871 links and 1260 commodities—the latter network was also used by Fontaine and Minner (2014) for solving the linearized link addition DNDP. To increase congestion effects on EM and BMC networks, we inflate trips by a factor of 4 and 2, respectively. Network information is summarized in Table 1.

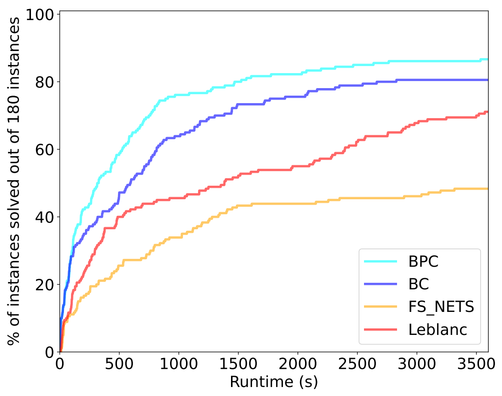
We use the SF DNDP instances introduced by Rey (2020) which consist of 20 instances: 10 with 10 additional new links, i.e. $|\mathcal{A}_2| = 10$ thus $|\mathcal{A}| = 86$; and 10 with 20 additional new links, i.e. $|\mathcal{A}_2| = 20$ thus $|\mathcal{A}| = 96$. For the EM and BMC networks, since these networks already have

Network (Acronym)	**Nodes**	**Links**	**Commodities**	**Trip inflation factor**
SiouxFalls (SF)	24	76	528	1
Eastern Massachusetts (EM)	74	258	1113	4
BerlinMitteCenter (BMC)	398	871	1260	2

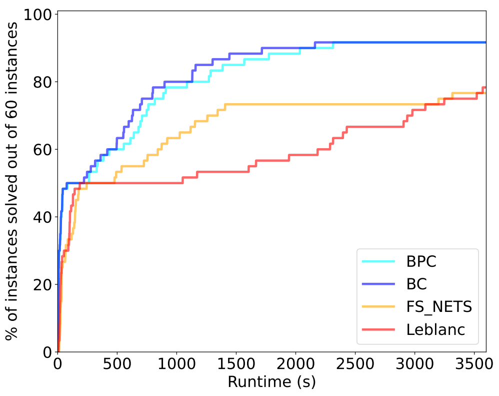
Table 1: Transportation networks used for generating DNDP instances.

many links, we generate DNDP instances by randomly sampling $|\mathcal{A}_2|$ links among existing links. For each sample, we verify the impact of closing these links one at a time by solving the corresponding UE-TAP and recording the TSTT percentage change relative to the original network. We discard samples if more than 1/3 of the sampled links do not generate an absolute change in TSTT greater than 1%. For both EM and BMC networks, we generate 20 DNDP instances: 10 with $|\mathcal{A}_2| = 10$ and 10 with $|\mathcal{A}_2| = 20$ and the cost of opening $\mathcal{A}_2$ is determined by randomly perturbing a linear function of links' free-flow travel time and capacity. For all three networks and for each of the 20 samples of $\mathcal{A}_2$ links and their costs, we generate three DNDP instances with a budget of 25%, 50% and 75% of the total cost of opening all links. This constitutes a dataset of a total of 180 DNDP instances including 60 instances of each transportation network. All algorithms are implemented in Python on a Windows machine with a i9 CPU at 3.19 GHz and 64 GB of memory. All LPs and MILPs are solved using CPLEX 22.1 MIP solver (International Business Machines Corporation 2024) with a single thread. We set an optimality gap of 1% for algorithm convergence and we set a runtime limit of 1 hour for each instance. For reproducibility, all data and codes are made available at `https://github.com/davidrey123/DNDP-path`.

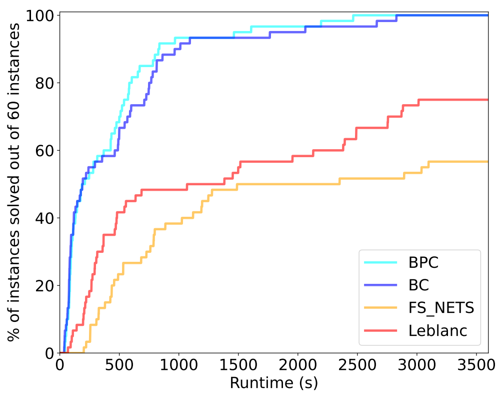
The main results of this study are reported in Figure 1 which depicts performance profile-like curves (Dolan and Moré 2002) of the four DNDP algorithms implemented, i.e. BPC, BC, FS_NETS and Leblanc. Specifically, for each algorithm, we report the percentage of instances solved over runtime. Figure 1a report performance profiles over all 180 instances considered. Performance profiles over the 60 instances of each network are reported in Figures 1b, 1c and 1d for SF, EM and BMC networks, respectively. This benchmarking reveals that, overall, BPC dominates all other three DNDP algorithms. It is able to solve to optimality over 50% of the instances within 315s while BC and Leblanc require 591s and almost 1383s to solve the same percentage of instances, respectively. Within the 1 hour runtime limit, BPC is able to solve 86.7% of the 180 instances whereas BC and Leblanc achieve 80.3% and 71.1%, respectively. In contrast, our implementation of FS_NETS is able to solve only 48.3% of the 180 instances considered within the runtime limit. A closer look at network-based performance sheds several insights. On SF instances, BC tends to slightly dominate BPC. We also find that FS_NETS dominates Leblanc for a significant range of runtimes. This highlights that, for small networks, the link-based multicommodity network flow model which is used in both BC and FS_NETS provides a viable alternative to path-based counterparts. EM instances reveal a different pattern: here FS_NETS is dominated by all other algorithms while BPC slightly dominates BC. This emphasizes the gains obtained by exploiting the separability of the leader objective function to generate OA cuts within the BC and BPC algorithms. Results on BMC instances—which is the largest network tested—demonstrate the substantial benefits of the BPC algorithm over the benchmarks considered. BPC is found to require 582s for solving 50% of these instances—which corresponds to 10-link BMC instances—and is able to solve 68.3% of BMC instances within the runtime limit. Leblanc is the second-best performing algorithm and requires 2096s to solve 50% of these instances. BC ranks third and requires 2545s to solve 50% of these



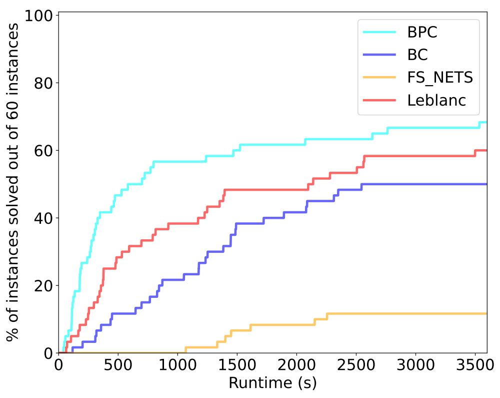
(a) All instances



(b) SF instances



(c) EM instances



(d) BMC instances

Figure 1: Performance profiles of DNDP algorithms: BPC, BC, FS_NETS and Leblanc.

instances; while FS_NETS only manages to solve 11.7% of these instances within the 1 hour runtime limit.

4 Conclusion

In this study, we presented a novel single tree branch-and-price-and-cut (BPC) algorithm for discrete network design problems under traffic equilibrium or DNDPs for short. DNDPs are notoriously challenging optimization problems which admit a natural Stackelberg game formulation in the presence of traffic equilibrium constraints. In this bilevel optimization formulation the leader represents the network designer while the follower represents a parameterized traffic equilibrium problem. In transportation, Wardrop's user equilibrium is often selected to model network users' route choice under congestion effects. Exploiting the separability and the convexity of the leader objective function, we introduce a new outer approximation (OA) scheme for the system-optimum (SO)-DNDP which corresponds to the high-point relaxation of the DNDP. Combining these successive relaxations of the bilevel problem with the path-based formulation of the DNDP, leads to a linear programming formulation that can be solved efficiently by column generation (CG). We develop a BPC framework to implement our approach and propose initialization techniques, cut generation rules and interdiction and value function cuts for algorithmic tuning.

We validate the performance of our BPC algorithm through comprehensive computational experiments over 180 problem instances based on three transportation networks of varying sizes. We use three alternative methods to compare the performance of the BPC algorithm including its branch-and-cut (BC) counterpart where a link-based multicommodity network model is used instead of the path-based network model. We show that our BPC algorithm outperform BC and existing approaches in the literature. Notably, we demonstrate that our BPC algorithm is efficient on both small and larger scale problem instances whereas other DNDP algorithms either fail to scale-up due to their reliance on link-based multicommodity network models (i.e. BC and FS_NETS) or to the inherent structure (Leblanc). In contrast, the BPC algorithm is able to consistently solve—or achieve competitive optimality gaps on—problem instances of varying number of variable links and/or network features. For reproducibility purposes, all data and codes used in this study are made publicly available at https://github.com/davidrey123/DNDP-path. For presentation and experimentation purposes, we focused on the link addition DNDP which is the most studied DNDP in the literature. We emphasize that most of the methods developed can be immediately applied to other DNDPs such as mixed discrete-continuous DNDPs or node-addition DNDPs.

This research can be extended in several directions. From a methodological standpoint, further research may explore the integration of additional cuts or penalty methods to reduce the optimality gap during search. While the value function cuts considered tend to reduce the number of BPC iterations, their incorporation leads to excessive computational efforts. Techniques to mitigate these effects could be explored. From a modeling perspective, this study focused on discrete NDPs, however the proposed OA relaxations and the CG approach can be applied to continuous NDPs as well. In this context, the branch-and-bound framework may be omitted and continuous NDPs could benefit from exploiting the OA and CG procedures developed in this study. DNDPs under traffic equilibrium have several practical applications notably in transportation networks but also in telecommunications networks (Correa and Stier-Moses 2011). Discrete NDPs arising in these contexts can benefit from the proposed approach by adapting its core elements to specific problem contexts and also extend to other discrete problems such as bilevel facility location or network operation scheduling problems under Wardropian equilibria.

References

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

Beckmann M, McGuire CB, Winsten CB (1956) Studies in the economics of transportation. Technical report.

Colson B, Marcotte P, Savard G (2007) An overview of bilevel optimization. *Annals of Operations Research* 153(1):235–256.

Correa JR, Stier-Moses NE (2011) Wardrop equilibria. *Encyclopedia of Operations Research and Management Science. Wiley* .

Dantzig GB, Wolfe P (1960) Decomposition principle for linear programs. *Operations Research* 8(1):101–111.

Dolan ED, Moré JJ (2002) Benchmarking optimization software with performance profiles. *Mathematical programming* 91:201–213.

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

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

Fletcher R, Leyffer S (1994) Solving mixed integer nonlinear programs by outer approximation. *Mathematical Programming* 66:327–349.

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

International Business Machines Corporation (2024) IBM ILOG CPLEX Optimization Studio. URL https://www.ibm.com/products/ilog-cplex-optimization-studio.

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

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

Magnanti TL, Wong RT (1984) Network design and transportation planning: Models and algorithms. *Transportation Science* 18(1):1–55.

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

Roughgarden T (2006) On the severity of Braess's paradox: Designing networks for selfish users is hard. *Journal of Computer and System Sciences* 72(5):922–953.

Sheffi Y (1985) *Urban transportation networks*, volume 6 (Prentice-Hall, Englewood Cliffs, NJ).

Transportation Networks for Research Core Team (2024) Transportation Networks for Research. URL https://github.com/bstabler/TransportationNetworks.

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

Wardrop JG (1952) Some theoretical aspects of road traffic research. *Inst Civil Engineers Proc London/UK/*.

Yang H, H Bell MG (1998) Models and algorithms for road network design: a review and some new developments. *Transport Reviews* 18(3):257–278.