On the properties of path additions for traffic routing

Matteo Bettini[a,*], Amanda Prorok[a]

[a]*Department of Computer Science and Technology, University of Cambridge, Cambridge, United Kingdom*

Abstract

In this paper we investigate the impact of path additions to transport networks with optimised traffic routing. In particular, we study the behaviour of total travel time, and consider both self-interested routing paradigms, such as User Equilibrium (UE) routing, as well as cooperative paradigms, such as classic Multi-Commodity (MC) network flow and System Optimal (SO) routing. We provide a formal framework for designing transport networks through iterative path additions, introducing the concepts of *trip spanning tree* and *trip path graph*. Using this formalisation, we prove multiple properties of the objective function for transport network design. Since the underlying routing problem is NP-Hard, we investigate properties that provide guarantees in approximate algorithm design. Firstly, while Braess' paradox has shown that total travel time is *not* monotonic non-increasing with respect to path additions under self-interested routing (UE), we prove that, instead, monotonicity holds for cooperative routing (MC and SO). This result has the important implication that cooperative agents make the best use of redundant infrastructure. Secondly, we prove via a counterexample that the intuitive statement 'adding a path to a transport network always grants greater or equal benefit to users than adding it to a superset of that network' is false. In other words we prove that, for all the routing formulations studied, total travel time is *not* supermodular with respect to path additions. While this counter-intuitive result yields a hardness property for algorithm design, we provide particular instances where, instead, the property of supermodularity holds. Our study on monotonicity and supermodularity of total travel

*Corresponding author

Email addresses: mb2389@cl.cam.ac.uk (Matteo Bettini), asp45@cl.cam.ac.uk (Amanda Prorok)

time with respect to path additions provides formal proofs and scenarios that constitute important insights for transport network designers.

Keywords: Transport network design, Supermodularity, Traffic routing.

---

1. Introduction

Traffic routing is a core transportation problem. It is concerned with directing multiple agents, with aligned or conflicting interests, to their destinations. Relevant applications include: road traffic [1], air traffic [2], crowd control [3], and internet packets [4]. With the advent of Connected Autonomous Vehicles (CAVs), we expect mobility paradigms to undergo a fundamental change, as we transition from (self-interested) human control to semi- (or fully) automated cooperative paradigms [5]. This will lead to a decrease in unpredictable routing behaviours caused by human decision making, paving the way for greater traffic efficiency [6, 7]. Despite of this shift from non-automated to semi- (or fully) automated traffic control, in most cases, the underlying infrastructure (e.g., transport network) remains the one that was originally designed for human control.

While many research efforts focus on developing algorithms for traffic planning and control, the transport network is often regarded as a fixed constraint. On the other hand, as we can observe from historic trends, every change to mobility systems was followed by a change to the transport network infrastructure [8]. We aim to anticipate this trend by investigating the properties of environment modifications under several (fixed) routing strategies.

The problem of optimizing the transport network topology is known as the Network Design Problem (NDP) [9]. This problem is generally NP-hard [10] and, thus, optimal solutions for large instances are computationally infeasible. Analysing the properties of the objective function, on the other hand, could provide us with additional tools for the design of approximation algorithms with known suboptimality bounds. In particular, for set functions that satisfy the property of diminishing returns, also known as sub/supermodularity, near-optimal approximations can be found [11, 12]. It is thus fundamental to provide an analysis of the properties of objective functions for network design under different routing objectives and paradigms.

In this work, we study the theory of transport network design through iterative path additions to derive and categorize its formal properties. We

Table 1: Summary of our results on the properties of total travel time with respect to path additions in transport networks. Multi-Commodity (MC) and System Optimal (SO) are cooperative routing formulations while User Equilibrium (UE) is self-interested.

Total travel time w.r.t. path additions	MC	SO	UE
Monotonic non-increasing	✓ Thm 3	✓ Thm 2	✗ Braess [13]
Supermodular (general case)	✗ Thm 4	✗ Thm 5	✗ Thm 5
Supermodular (special case)	✓ Thm 6	✓ Thm 7	✓ Thm 7

model traffic as flow traversing a transport network represented as a graph. Under this formulation, we consider three different routing problems tasked with guiding each agent to its destination: *i) minimum cost multi-commodity network flow problem (MC)* [14], where agents experience constant travel times, *ii) system optimal (SO) routing* [15], where travel time is dependent on edge congestion and agents cooperate to minimize the systems' cost (total travel time), and *iii) user equilibrium (UE) routing* [15], also known as Wardrop equilibrium [16], where travel time is dependent on edge congestion and agents are routed to minimize their individual travel time. These routing paradigms are coordinated as they all require a centralised routing infrastructure, following three different optimisation objectives. We furtherly refer to MC and SO as *cooperative* routing, as they optimise for the system's benefit, and to UE as *self-interested* routing, as it optimises for individual agents. We are interested in analysing the properties of total travel time for these routing problems under path additions to the transport network. We thus introduce the concepts of *trip spanning tree* and *trip path graph* which allow us to formalise what we consider an initial transport network and the space of its possible path additions. This model is then leveraged to prove our properties of interest.

Firstly, while Braess [13] has shown that total travel time is not monotonic non-increasing with respect to path additions under self-interested UE routing, we prove in our formulation that this does *not* occur for cooperative routing (MC and SO). This result has the important implication that

cooperative agents make the best use of redundant paths in the network.

Secondly, we prove that the intuitive statement 'adding a path to a transport network always grants greater or equal benefit to users than adding it to a superset of that network' is false. In other words, we prove that total travel time is not supermodular with respect to path additions. This is valid both when agents are routed according to their own interest (UE) and when they are cooperating for the system's benefit (MC and SO). We provide a counterexample to support our proofs. While this counter-intuitive result states that the supermodularity of path additions cannot be leveraged in approximation algorithms for general network optimisation, we provide particular instances where, instead, the property of supermodularity holds. In this scenario only parallel path additions are allowed, making it suitable to represent network design settings where physical additions can overlap but virtual network paths are kept parallel (e.g., by creating dedicated lanes).

Our results are summarised in Table 1. This work contains the following contributions:

- We present the problem of network design via iterative path additions and we introduce a graph structuring framework for network design which leverages the new concepts of *trip spanning tree* and *trip path graph*. Braess' example [13] fits in this framework.
- We prove that, for cooperative agents, the total travel time is monotonic non-increasing with respect to path additions.
- We prove with a counterexample that, in the general case, the total travel time is *not* supermodular with respect to path additions for both cooperative and self-interested agents.
- We introduce and prove a special case where supermodularity holds (i.e., restricted supermodularity) and can thus be leveraged for designing approximation algorithms for specific network design instances.

The paper is structured in the following way: Section 2 goes through the relevant related work, Section 3 discusses some preliminaries, in Section 4 our formulation is introduced, Section 5 goes through the routing paradigms used, Section 6 describes our formalization of the initial transport network and its extension via path additions, Section 7 investigates the properties of total travel time with respect to path additions, discussing monotonicity and

non-supermodularity, to then focus on particular supermodular instances, and, finally, Section 8 concludes the paper.

2. Related work

The problem of designing transport networks for traffic routing has been studied for a long time and goes by the name of Network Design Problem (NDP). It is also known as the Road Network Design Problem (RNDP) [17] and Transport Network Design Problem (TNDP) [18]. Extensive discussion on the topic can be found in [19, 10, 17]. The problem is usually formulated as a bilevel optimisation task. At the higher level, the network designer can make decisions that impact the network infrastructure. These decisions can be discrete (e.g., adding and removing edges) or continuous (e.g., extending the edge capacities). The objective of the network designer is usually to minimise the total travel time and, thus, the network congestion. Decisions involve physical network modifications which can be subject to a construction budget. At the lower level, agents see the network modifications and distribute in the resulting network according to their routing objective. Note that the designer cannot control directly the agents' behaviour, but only influence it through environment optimisation. The designer's goal is to minimise the total system travel time while respecting the construction budget. In other words, make the best use of network modifications in order to improve the routing performance according to the routing objective of choice. The NDP can also be viewed in a game theoretic framework, where the bilevel task is a Stackelberg leader-follower game [20] with agents in the lower level playing a cooperative (MC, SO) or non-cooperative (UE) game depending on the routing objective.

The nature of the decision variables at the higher level divides the NDP into the Continuous Network Design Problem (CNDP), which considers continuous network modifications, and the Discrete Network Design Problem (DNDP), which considers discrete network modifications. Typical decision variables for the CNDP are: road capacities [21, 22, 23], traffic light schedules [24, 25], and road tolls [26, 27]. Typical decision strategies for the DNDP are: new road construction [9, 28, 29], lane allocation [30, 31], and road capacity expansion [32, 33]. In this work, we introduce a novel formulation for the higher level of the DNDP, where we consider adding paths to a transport network. Therefore, the space of possible path additions constitutes our decision variables. The addition of a path can lead to the construction of zero

or multiple new roads.

At the lower optimisation level, most existing works consider agents routed according to some form of User Equilibrium (UE) routing: a formulation that characterizes a non-cooperative game in which every agent is self-interested and chooses the fastest path for themself according to the current congestion, leading to a Nash equilibrium. Popular variations of UE are: stochastic UE [34] which considers uncertainty in the travel time and dynamic UE [35] which considers dynamic user demands. The choice of UE for routing in the NDP literature is motivated by its fairness (agents travelling from the same origin to the same destination experience the same travel time) and by the intrinsic self-interested nature of humans. On the other hand, we argue that, with the introduction of CAVs, we will see a shift towards cooperative routing, as human self-interest will be replaced by autonomous cooperation. One example of this phenomenon is the recent introduction of autonomous mobility-on-demand vehicles [36], which, by cooperating through a central routing authority, are able to reduce traffic congestion. In the literature, few works use System Optimal (SO) routing for the lower level [37, 38]. Motivated by these considerations, in this work we provide results for both cooperative (MC, SO) and self-interested (UE) routing paradigms.

Our problem is an instance of the DNDP. The DNDP is generally NP-Hard [10]. Its hardness is caused by a series of factors. Firstly, even the most simple bilevel problem with linear upper and lower levels is strongly NP-Hard [39, 37]. Secondly, even if both levels are convex, the convexity of the bilevel problem cannot be guaranteed [40]. Lastly, the discrete formulation of our decision variables highlights the combinatorial nature of the problem. Therefore, it is computationally intractable to solve the problem exactly for large instances. A branch-and-bound algorithm and a SO relaxation for the lower bounds is proposed in [9]. Later, another branch-and-bound solution for agents routed using stochastic UE is introduced by [41]. However, these exact methods are computationally infeasible for large networks and, thus, some alternatives have been explored. Branch and backtrack heuristics have been used in [32], while metaheuristic hybrids based on ant colony optimisation are proposed in [33, 42]. The genetic algorithm represents one effective metaheuristic solution [43], which has also been applied to the CNDP [44, 45].

Although metaheuristics achieve supposedly good solutions, they do not provide suboptimality guarantees. This is a main obstacle for the adoption of such network design solutions by companies and traffic regulators [46]: the absence of guarantees lowers the marketability of heuristics and their

unknown performance undermines the users' trust. Furthermore, without an available optimal solution, the goodness of metaheuristics cannot even be measured a posteriori. On the other hand, by leveraging certain properties of the objective function, it is possible to design approximation algorithms with suboptimality bounds. Such algorithms present all the characteristics required by companies and traffic regulators: they are computationally efficient while providing goodness guarantees. In particular, for set functions that satisfy the property of diminishing returns, also known as sub/supermodularity, near-optimal approximations can be found [11]. There exist many works that exploit sub/supermodularity in particular instances of network design. An efficient topology optimisation algorithm for information diffusion is designed in [47]. In [48], submodularity is leveraged to achieve near-optimal sensor placement in traffic networks. Submodular network design to achieve desired algebraic rigidity properties is explored in [49]. Restricted supermodularity for robust network design is proved in [50]. While these works are able to exploit sub/supermodularity for network design, they all leverage particular problem formulations designed for their specific instance. Instead, in this work, we provide results on the properties of objective functions for network design both in general and special cases under multiple routing formulations in order to provide a comprehensive analysis.

3. Preliminaries

We now introduce some preliminary concepts in transportation research and set functions. In Section 3.1, we discuss the Braess' paradox [13], which proves via counterexample that travel time of self-interested agents is not monotonic non-increasing with respect to path additions. This work relates to our study as we investigate the same property for cooperative agents. In Section 3.2, we introduce supermodularity, a fundamental property of set functions leveraged in approximation algorithm design [11]. This property will be discussed in the context of transport network path additions in Section 7.

3.1. Braess' paradox

One important result in traffic research is a routing paradox discovered by Braess [13] in 1968, which has since become known as the Braess' paradox. Braess' paradox states that, when users are routed according to UE, the total system travel time can increase after a new path is added to the network.

In other words, it presents a proof that the total travel time of users routed in a network according to UE is not monotonic non-increasing with respect to path additions. We can see how this phenomenon is not really a paradox as, when users are behaving according to UE, they behave according to self-interest and thus probably do not make the best use of the transportation infrastructure available. Braess illustrates this through a simple yet somewhat pathological example. However, Braess' paradox has been shown to be commonplace in road networks [51, 52] and in large random networks [53].

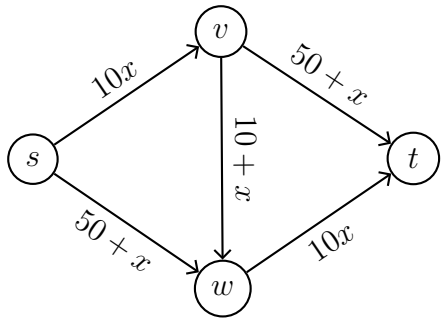


Figure 1: The Braess' paradox network. Labels on the edges represent the flow-dependant travel time functions. This network shows that, when a flow of 6 must be routed from s to t according to UE, the total travel time increases after the addition of edge (v, w).

In Braess' example, shown in Fig. 1, we consider a flow demand of 6 going from node s to node t. The road travel time cost functions are reported on the respective edges of the graph. In the case where (v, w) is not present in the network, the solutions to SO and UE routing coincide: all the users spread equally on the available paths, leading to a flow of 3 on the top edges and a flow of 3 on the bottom edges. Every user experiences the same travel time of $50 + 3 + 10 * 3 = 83$. This solution is both optimal for the users and for the system. On the other hand, if the edge (v, w) is added to the network, the solution to the SO does not change, while self-interested users behaving according to UE see an opportunity to decrease their travel time by using the newly created path $(s, v) \rightarrow (v, w) \rightarrow (w, t)$. Thus, when the equilibrium is reached, we observe a flow of 2 in the previously available paths and a flow of 2 in the newly created one. Note that the flow on an edge is the sum of the flows on the paths passing through it, thus edge (s, v) will have a flow of 4. In this UE scenario each user experiences the same travel time of 92. The

increase in the total travel time of users behaving accordingly to UE after the addition of edge (v, w) is: $92 * 6 - 83 * 6 = 552 - 498 = 54$.

Braess' paradox informs us that extreme cation must be used when structuring transport networks, as it presents a counterexample to prove that total travel time is not monotonic with respect to path additions under UE routing. Our work extends this result by proving that total travel time is not supermodular with respect to path additions under either MC, SO or UE routing.

3.2. Supermodularity

A fundamental component underlying this work is the property of supermodularity of set functions [54]. Before introducing such property, let us first define non-increasing monotonicity.

Definition 1 (Monotonic non-increasing set function)**.** *Let Λ be a set function defined as $\Lambda : 2^{\mathcal{G}} \mapsto \mathbb{R}$, where $2^{\mathcal{G}}$ is the power set of the finite set $\mathcal{G}$. Λ is a monotonic non-increasing function of $\mathcal{G}$ if and only if for every $\mathcal{A}$ and $\mathcal{B}$ such that $\mathcal{A} \subseteq \mathcal{B} \subseteq \mathcal{G}$, we have that $\Lambda(\mathcal{B}) \leqslant \Lambda(\mathcal{A})$.*

Simply put, if Λ is monotonic non-increasing, its output cannot increase when new elements are added to its current input. Let us now introduce supermodularity.

Definition 2 (Supermodular set function)**.** *Let Λ be a set function defined as $\Lambda : 2^{\mathcal{G}} \mapsto \mathbb{R}$, where $2^{\mathcal{G}}$ is the power set of the finite set $\mathcal{G}$. Λ is a supermodular function of $\mathcal{G}$ if and only if for every $\mathcal{A}$ and $\mathcal{B}$ such that $\mathcal{A} \subseteq \mathcal{B} \subseteq \mathcal{G}$ and every $x \in \mathcal{G} \setminus \mathcal{B}$, we have that*

$$\Lambda(\mathcal{A}) - \Lambda(\mathcal{A} \cup \{x\}) \geqslant \Lambda(\mathcal{B}) - \Lambda(\mathcal{B} \cup \{x\}).$$

In other words, if Λ is supermodular and it represents some notion of cost, the utility of adding an element to its current input is always greater or equal than the utility of adding the same element to a superset of its current input.

It is fundamental to understand that none of these two properties implies the other. Therefore, a supermodular function can be non-monotonic.

4. Problem formulation

In this work, we study traffic on transport networks as flow traversing a graph. We denote each commodity (flow origin-destination pair) as a trip. The source and destination represent the extremities of the trip, and the demand represents the number of vehicles per time unit wishing to go from that origin to that destination. The edge cost represents the travel time. *From this point on we will use the terms commodity and trip interchangeably.*

Given an oriented graph $G = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ denotes the node set and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ denotes the edge set, and a set of trips as $\mathcal{M} = \{(s^m, t^m, d^m)\}_m$, where $s^m \in \mathcal{V}$ is the origin of trip m, $t^m \in \mathcal{V}$ is the destination, and $d^m \in \mathbb{R}_{>0}$ is the trip demand. We associate to each edge $(i,j) \in \mathcal{E}$ a cost $c_{ij} \in \mathbb{R}_{>0}$ representing the travel time and a capacity $u_{ij} \in \mathbb{R}_{\geqslant 0}$ representing the maximum flow sustainable. Note that, in this formulation, we allow $c_{ij} \neq c_{ji}$. The cost can be a scalar or a function, depending on the routing formulation (see Section 5). Each node $i \in \mathcal{V}$ has a demand $d_i^m \in \mathbb{R}, \forall m \in \mathcal{M}$. We set the network demands d_i^m as indicated by:

$$\forall i \in \mathcal{V},\ \forall m \in \mathcal{M} \quad d_i^m = \begin{cases} -d^m & \text{if } i = s^m \\ d^m & \text{if } i = t^m \\ 0 & \text{otherwise.} \end{cases}$$

We observe that the total demand of the transport network is equal to zero $\sum_{i \in \mathcal{V}} \sum_{m \in \mathcal{M}} d_i^m = 0$. We represent the vehicle flow of trip $m \in \mathcal{M}$ on edge $(i,j) \in \mathcal{E}$ as $x_{ij}^m \in \mathbb{R}_{\geqslant 0}$, where $\sum_{m \in \mathcal{M}} x_{ij}^m = x_{ij}$ is the total flow on edge (i,j). A summary of the notation used in the paper is available in Appendix A.

We are interested in analysing the properties of routing under path additions. Therefore, to complement our problem formulation, in the next two sections we introduce the routing paradigms used (Sec. 5) and how we formalise path additions (Sec. 6).

5. Background on coordinated routing

We now introduce the three coordinated routing formulations considered in this work: MC (Sec. 5.1), SO (Sec. 5.3), and UE (Sec. 5.4). We refer to all of them as coordinated as they require a central authority to compute routing, but we furtherly denote MC and SO as *cooperative*, as vehicles are routed for the system's benefit and UE as *self-interested*, as vehicles are routed according to their own interest.

5.1. Minimum cost multi-commodity network flow problem

The minimum cost multi-commodity network flow problem (MC) is a classic flow problem [14]. It is concerned with capacitated networks in which a constant cost is associated to every edge. Given a set of trips, each with a flow demand to be routed from an origin node to a destination node, we are interested in finding the optimal flow distribution in the network such that the total cost is minimised. The shortest path problem and the maximum flow problem are special cases of the MC problem.

The routing problem consists in determining the optimal agent distribution on the network edges in order to route all the demand d^m from s^m to t^m, $\forall m \in \mathcal{M}$ while minimising the total cost. The problem is formalised as follows:

$$\min_{\{x_{ij}^m\}} \sum_{m \in \mathcal{M}} \sum_{(i,j) \in \mathcal{E}} x_{ij}^m c_{ij} \tag{1a}$$

s.t.

$$\sum_{m \in \mathcal{M}} x_{ij}^m \leqslant u_{ij} \quad \forall (i,j) \in \mathcal{E} \tag{1b}$$

$$x_{ij}^m \geqslant 0 \quad \forall (i,j) \in \mathcal{E},\ \forall m \in \mathcal{M} \tag{1c}$$

$$\sum_{\{j:(j,i) \in \mathcal{E}\}} x_{ji}^m - \sum_{\{j:(i,j) \in \mathcal{E}\}} x_{ij}^m = d_i^m \quad \forall i \in \mathcal{V},\ \forall m \in \mathcal{M}, \tag{1d}$$

where the objective function (1a) minimises the total travel time over all trips. Constraint (1b) ensures that the sum of the flows from different trips on each edge does not exceed the maximum capacity. Constraint (1c) ensures that no flow is negative. Constraint (1d) ensures flow conservation.

Path formulation. According to the flow decomposition theorem, first introduced by [55], we can reformulate the MC problem using the distribution of flow over paths instead of edges. This is valid under the assumption that for every trip there exist no negative cost cycles in our graph.

For every trip $m \in \mathcal{M}$, let P^m denote the set of all possible paths[1] from s^m to t^m and $P = \bigcup_{m \in \mathcal{M}} P^m$ the union of such sets. The constant cost c_p of a path is defined as the sum of the constant cost of its edges $c_p = \sum_{(i,j) \in p} c_{ij}$.

[1] A path is a is a walk in which all vertices (and therefore also all edges) are distinct.

We also define as x_p the flow on path $p \in P^m$ and as $\mathbf{x}$ the vector of all path flows x_p, $\forall p \in P$. The path formulation goes as follows:

$$\min_{\mathbf{x}} \sum_{p \in P} x_p c_p \tag{2a}$$

s.t.

$$\sum_{p \in P:(i,j) \in p} x_p \leqslant u_{ij} \quad \forall (i,j) \in \mathcal{E} \tag{2b}$$

$$x_p \geqslant 0 \quad \forall p \in P \tag{2c}$$

$$\sum_{p \in P^m} x_p = d^m \quad \forall m \in \mathcal{M}, \tag{2d}$$

where the objective function and the constraints map exactly to the ones illustrated in the edge formulation of MC.

5.2. Flow-dependant cost

In reality, it is often not possible to map the travel time of road segments to a constant cost. In fact, *the travel time required to traverse a road is highly dependent on its congestion*. It can be modeled by a continuous monotonic increasing function mapping the current flow on an edge to travel time. In this work, we will consider two popular travel time functions used in traffic research, but our theoretical results remain valid for any reasonable convex travel time function.

5.2.1. Greenshields affine velocity function

This function was introduced by Greenshields in 1935 [56] and still remains a useful mathematical model thanks to its simplicity. It is well-known in traffic simulation research [57, 58]. It is defined as follows:

$$c_{ij}(x_{ij}) = \frac{l_{ij}}{v_{ij}^{\max}\left(1 - \frac{x_{ij}}{u_{ij}}\right)}, \tag{3}$$

where $v_{ij}^{\max}$ is the maximum velocity allowed on edge (i,j) and l_{ij} is the length of (i,j). Note that the input variable x_{ij} has to be a feasible assignment and thus $0 \leqslant x_{ij} \leqslant u_{ij}$. The function models a convex hyperbolic relation between congestion and travel time.

5.2.2. The Bureau of Public Roads (BPR) cost function

Another frequently considered convex latency function is the Bureau of Public Roads (BPR) cost function [59, 60]. The function is computed as

$$c_{ij}(x_{ij}) = c_{ij}^0 \left(1 + \alpha_{ij} \left(\frac{x_{ij}}{u_{ij}}\right)^{\beta_{ij}}\right), \tag{4}$$

where c_{ij}^0 and u_{ij} are the free-flow time and capacity of link (i, j), respectively, and α_{ij} and β_{ij} are shape parameters which can be calibrated to data. By tuning these parameters a wide variety of functions can be represented, including the ones discussed in Section 3.1 used by Braess to exacerbate his paradox [13].

5.3. System optimal (SO) routing

Now we can use the cost functions just introduced to extend our formulation of the MC problem. Each edge $(i, j) \in \mathcal{E}$ is assigned the flow-dependant cost function $c_{ij}(x_{ij})$. The vector of edge cost functions $c_{ij}(x_{ij})$ is noted as $\mathbf{c}$. We call the triple $(G, \mathcal{M}, \mathbf{c})$ an instance. The cost of a path $p \in P$ with respect to a flow x_p is defined as the sum of the cost of the edges in the path $c_p(x_p) = \sum_{(i,j)\in p} c_{ij}(x_p)$.

The optimal flows are then characterised by the following System Optimal (SO) [15] routing problem:

$$\min_{\mathbf{x}} \sum_{p\in P} x_p c_p(x_p) \tag{5a}$$

s.t.

$$x_p \geqslant 0 \quad \forall p \in P \tag{5b}$$

$$\sum_{p\in P^m} x_p = d^m \quad \forall m \in \mathcal{M}, \tag{5c}$$

where constraint (5b) states that all flows must be non-negative and constraint (5c) imposes the conservation of flow for each trip. Note that the SO problem corresponds to the MC problem with flow-dependant travel time functions and without the strict edge capacity constraints.

Since the local and global optima of a convex function on a convex set coincide [61], we know that a locally optimal solution for SO is also globally optimal whenever the objective function (5a) is convex. By construction, we

consider only convex cost functions, like the ones introduced in Section 5.2. The path cost $c_p(x_p)$ and the SO objective are thus also convex, since a non-negative weighted sum of convex functions is still convex. From [15] we know that a flow is locally (and globally) optimal if and only if moving flow from one path to another can only increase the global cost (5a). That is, we expect to have optimality when the marginal benefit of decreasing flow on a $s^m - t^m$ path is at most the marginal cost of increasing flow on another $s^m - t^m$ path. The following lemma formalises this intuition. Let us denote the effective cost $k_{ij}(x_{ij}) = x_{ij}c_{ij}(x_{ij})$. We write the derivative of k_{ij} as k'_{ij} and define $k'_p(x_p) = \sum_{(i,j)\in p} k'_{ij}(x_p)$. We then have:

Lemma 1 ([62, 63, 15])**.** *The flow vector* $\mathbf{x}$ *computed by SO routing on an instance* $(G, \mathcal{M}, \mathbf{c})$ *is optimal if and only if for every* $m \in \mathcal{M}$ *and any pair* $p_1, p_2 \in P^m$ *with* $x_{p_1} > 0$*, we have* $k'_{p_1}(x_{p_1}) \leqslant k'_{p_2}(x_{p_2})$*.*

The solution to the SO routing problem thus characterises the scenario where all users choose their paths in order to minimise the total system travel time or, in other words, the social cost.

5.4. User equilibrium (UE) routing

By solving the SO problem we optimise for total system travel time. On the other hand, some agents could experience longer travel times than others on the same trip for the system's benefit. Due to this characteristic, we define SO as *unfair*.

A fair assignment is the User Equilibrium (UE). It induces a Nash equilibrium in which no user can improve their travel time (cost) by choosing a different route. It is also known as Wardrop equilibrium [16]. This equilibrium emerges when we consider the routing problem as a non-cooperative game, where each agent, seen as a fraction of the flow, is routed on the fastest route available according to the current traffic conditions. In this sense, agents routed according to UE can be defined as self-interested.

Lemma 2 ([15])**.** *A feasible flow* $\mathbf{x}$ *for instance* $(G, \mathcal{M}, \mathbf{c})$ *is at UE if and only if for every* $m \in \mathcal{M}$ *and any pair* $p_1, p_2 \in P^m$ *with* $x_{p_1} > 0$*, we have* $c_{p_1}(x_{p_1}) \leqslant c_{p_2}(x_{p_2})$*.*

In particular, if a flow is at UE, all the paths of trip m for which $c_p > 0$ have the same cost C_m.

Lemma 3 ([15]). *If a flow is at UE for instance* $(G, \mathcal{M}, \mathbf{c})$*, then the total cost is computed as*

$$\sum_{p \in P} x_p c_p(x_p) = \sum_{m \in \mathcal{M}} d^m C_m. \tag{6}$$

This means that all agents on the same trip will arrive at the same time to their destination under UE. The convexity guarantees of SO are valid also in this case and, in particular, we have strong existence guarantees for the equilibrium.

Lemma 4 ([15]). *Given an instance* $(G, \mathcal{M}, \mathbf{c})$*, if there exists a feasible flow assignment, there always exists a feasible flow assignment at UE. Furthermore, given two assignments at UE, they will have the same total cost (travel time) (shown in Eq.* (6)*).*

Lastly, let us define the UE routing problem as follows:

$$\min_{\mathbf{x}} \sum_{(i,j) \in \mathcal{E}} \int_0^{x_{ij}} c_{ij}(\tau) d\tau \tag{7a}$$

s.t.

$$x_p \geqslant 0 \quad \forall p \in P \tag{7b}$$

$$\sum_{p \in P^m} x_p = d^m \quad \forall m \in \mathcal{M} \tag{7c}$$

$$x_{ij} = \sum_{p \in P : (i,j) \in p} x_p \quad \forall (i,j) \in \mathcal{E}, \tag{7d}$$

where the constraints are the same as in the SO case, with the addition of (7d), which states that the flow on an edge is equal to the sum of the flows of the paths passing through it. This addition is necessary as in the objective (7a) we sum over edges instead of paths.

5.5. Differences between SO and UE routing

The SO and UE problems are different. In particular, when considering the UE formulation, we trade-off total travel time for fairness and self-interest.

However, by looking at Lemmas 1 and 2, we can notice some similarities. The striking similarity between the characterizations of optimal solutions to the SO and UE problems was noticed early on [62], and provides an interpretation of a system optimal flow as a flow at UE with respect to a different set of edge cost functions [15]. To make this relationship precise, denote the marginal cost of increasing flow on edge (i, j) by $c^*_{ij}(x_{ij}) = k'_{ij}(x_{ij}) = (x_{ij}c_{ij}(x_{ij}))' = c_{ij}(x_{ij}) + x_{ij}c'_{ij}(x_{ij})$. Lemmas 1 and 2 yield the following corollary.

Corollary 1 ([15])**.** *Let $(G, \mathcal{M}, \mathbf{c})$ be an instance and $\mathbf{c}^*$ be the vector of marginal cost functions defined as above. Then, a flow feasible for $(G, \mathcal{M}, \mathbf{c})$ is SO if and only if it is at UE for the instance $(G, \mathcal{M}, \mathbf{c}^*)$.*

Both routing strategies represent convex nonlinear optimisation problems, which can be solved by traditional gradient-based techniques such as the Frank-Wolfe algorithm [64, 65]. The cost trade-off in total travel time between SO and UE is known as "The price of anarchy" [66, 67]. Note that, in the MC problem, due to the constant costs, the optimal solution is guaranteed to be always at SO and UE (assuming we ignore the capacity constraint (1b)).

Example 1. Let us consider the example shown in Fig. 2, first introduced by Pigou [68] to illustrate the difference between SO and UE routing. In this simple scenario we consider only one trip, from s to t, with a demand of 1 unit of flow. In the network there are only two paths, represented as edges. The travel time of such paths is indicated by $c_1(x_1)$ and $c_2(x_2)$, which are functions of the flow. The flow routed on the upper road is indicated as x_1, while the flow on the lower road is x_2.

If the users are routed according to UE, the flow will distribute such that all the users experience the same travel time. This leads to a UE where all the flow is routed on the lower road ($x_1 = 0, x_2 = 1$), yielding $c_1(x_1) = c_2(x_2) = 1$. The total system travel time in this case is $x_1c_1(x_1) + x_2c_2(x_2) = 1$. On the other hand, if we optimise for the SO, the flow distributes equally on the two roads ($x_1 = 0.5, x_2 = 0.5$), yielding a total travel time of $x_1c_1(x_1)+x_2c_2(x_2) = 0.75$. This example is summarised in Table 2.

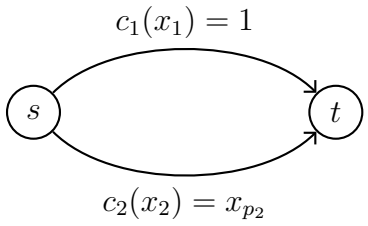


Figure 2: Pigou's example to illustrate the difference between UE and SO. It presents one origin s and one destination t and a flow demand of 1 that has to be distributed along paths 1 and 2, with travel time costs respectively $c_1(x_1)$ and $c_2(x_2)$. While SO distributes the flow equally among the two paths, UE routes all the flow on the lower path to guarantee equal costs on the two paths.

Table 2: UE and SO routing results on the network in Fig. 2.

Routing	x_1	x_2	Total travel time $x_1c_1(x_1) + x_2c_2(x_2)$
User Equilibrium	0	1	1
System Optimal	0.5	0.5	0.75

6. Path additions

Now that we have introduced the routing problems of interest, we turn our attention to the transport network on which vehicles are routed, represented as a graph. We are interested in developing a formulation for the incremental construction of the graph which enables us to formally reason about the properties of path additions. We introduce two concepts: the *trip spanning tree* and the *trip path graph*. These two components allow us to simply model path additions in a modular framework, which is also able to represent the example introduced by Braess' [13].

We consider graphs of the type introduced in Section 4. All the graphs we will consider, will be subgraphs of a *template graph* $G_{\mathcal{T}}$. This template graph is used to limit the space of possible path additions. In this work we consider the template graph to be a square grid lattice with bidirectional edges, but any template choice is equally valid.

As we are interested in analysing the properties of path additions to a transport network, we have to start by defining an initial feasible graph

which we aim to extend. Therefore, we start by introducing the concept of *trip spanning tree*.

Definition 3 (**Trip spanning tree**). *Given a template graph $G_{\mathcal{T}}$ and a set of trips $\mathcal{M} = \{(s^m, t^m, d^m)\}_m$, a trip spanning tree is a directed graph $G_{\mathcal{I}} = (\mathcal{V}_{\mathcal{I}}, \mathcal{E}_{\mathcal{I}})$, subgraph of $G_{\mathcal{T}}$, with the following four properties:*

1. $s^m, t^m \in \mathcal{V}_{\mathcal{I}},\ \forall m \in \mathcal{M}$
2. $|P^m| = 1,\ \forall m \in \mathcal{M}$
3. $\forall n \in \mathcal{V}_{\mathcal{I}},\ \exists m \in \mathcal{M},\ p \in P^m : n \in p$
4. $\sum_{m \in \mathcal{M}} \sum_{p \in P^m : (i,j) \in p} d^m \leqslant u_{ij} \quad \forall (i,j) \in \mathcal{E}_{\mathcal{I}}$.

Property (1) states that a trip spanning tree must contain the source and destination of every trip. Property (2) states that for each origin-destination pair there exists exactly one path connecting them in the trip spanning tree. Property (3) states that every node in the trip spanning tree must be part of a path connecting one trip's origin to its destination. Property (4) states that the capacity of the edges in the trip spanning tree must be sufficient to guarantee a feasible solution to the MC problem. These properties are designed to make the trip spanning tree an initial feasible graph for the routing problem. Note that the trip spanning tree is not unique with respect to the problem instance.

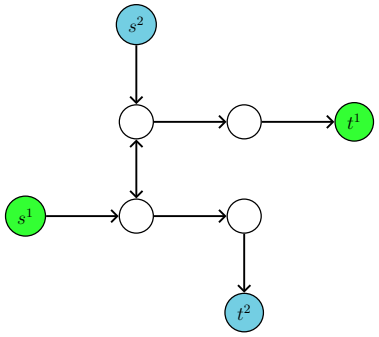
Let us now define the possible path additions through the concept of *trip path graph*.

Definition 4 (**Trip path graph**). *Given a template graph $G_{\mathcal{T}}$ and a set of trips $\mathcal{M} = \{(s^m, t^m, d^m)\}_m$, we denote a trip path graph, with respect to trip $m \in \mathcal{M}$, as a graph $G_{m_x} = (\mathcal{V}_{m_x}, \mathcal{E}_{m_x})$, subgraph of $G_{\mathcal{T}}$, with the following properties:*

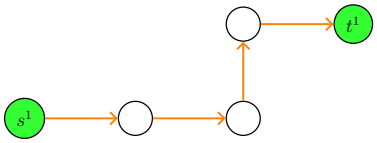
1. $s^m, t^m \in \mathcal{V}_{m_x}$
2. $|P^m| = 1$
3. $\forall n \in \mathcal{V}_{m_x},\ p \in P^m : n \in p$.

A trip path graph is thus a path graph which contains only a path from one trip's source to its destination. The number of possible trip path graphs for trip m is dependent on the template graph $G_{\mathcal{T}}$ and is indexed by x. Property (1) states that the origin and destination of such trip must belong to the trip path graph. Property (2) states that the trip path graph must contain only one path between these two. Property (3) states that all the

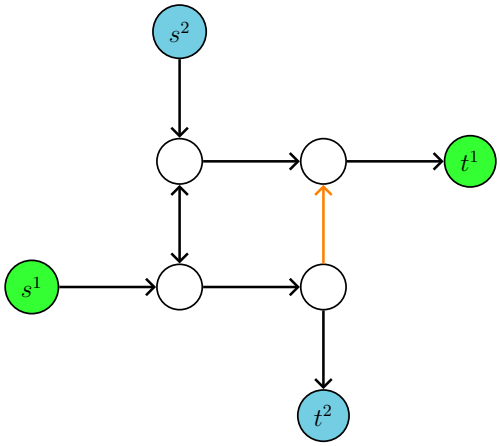
nodes belonging to the trip path graph must belong to such path. Therefore, a trip path graph is a trip spanning tree with respect to only one trip, without the guarantees given by Property (4) of a trip spanning tree. Note that the Braess paradox (Fig. 1) can be formulated according to this framework. Here a trip path graph (s, v, w, t) is added to a trip spanning tree that has already undergone one path addition.



(a) Trip spanning tree $G_{\mathcal{I}}$



(b) Trip path graph G_{1_1}



(c) Graph union $G_{\mathcal{I} \oplus 1_1}$

Figure 3: A simple transport network with two trips to illustrate graph unions. Here we have a trip spanning tree (a) and a trip path graph for trip 1 (b) that are being unified in (c).

The trip spanning tree and the trip path graphs allow us to model the subsequent addition of paths to an initial graph. We describe this through graph unions. We denote the graph obtained by adding a trip path graph $G_{m_x} = (\mathcal{V}_{m_x}, \mathcal{E}_{m_x})$ to the trip spanning tree $G_{\mathcal{I}} = (\mathcal{V}_{\mathcal{I}}, \mathcal{E}_{\mathcal{I}})$ as $G_{\mathcal{I} \oplus m_x} =$

$(\mathcal{V}_{\mathcal{I}} \cup \mathcal{V}_{m_x}, \mathcal{E}_{\mathcal{I}} \cup \mathcal{E}_{m_x})$. Note that, by this addition, the trip path graph can introduce multiple new nodes and edges not previously contained in the trip spanning tree and thus also multiple new paths for each trip. The number of new paths introduced is limited only by the template graph $G_{\mathcal{T}}$ (remembering that $G_{\mathcal{I}} \subseteq G_{\mathcal{T}}$ and $G_{m_x} \subseteq G_{\mathcal{T}}$).

We illustrate the concept of trip spanning tree and trip path graph addition with an example, shown in Fig. 3. In this simple example we have only two trips $\mathcal{M} = \{(s^1, t^1, d^1), (s^2, t^2, d^2)\}$. We instantiate a possible trip spanning tree $G_{\mathcal{I}}$ (Fig. 3a) and a possible trip path graph G_{1_1} (Fig. 3b) for trip 1. The resulting graph union $G_{\mathcal{I} \oplus 1_1}$ is shown in Fig. 3c. Despite G_{1_1} has four edges, only one new edge is added to the graph, since the other edges were already present. Furthermore, note that, in this case, G_{1_1} adds one path for trip 1 and no paths for trip 2. This is not always the case as adding a trip path graph for trip m could introduce new paths for other trips as well.

Let us introduce some further notation related to graph unions. We consider a graph $G_{\mathcal{A}} = (\mathcal{V}_{\mathcal{A}}, \mathcal{E}_{\mathcal{A}})$, obtained after $n \geqslant 0$ trip path graph additions to a trip spanning tree, and a trip path graph $G_{m_x} = (\mathcal{V}_{m_x}, \mathcal{E}_{m_x})$. We denote with $P^m_{\mathcal{A}}$ the set of all paths for trip m in $G_{\mathcal{A}}$, with $P^m_{m_x|\mathcal{A}}$ the set of all paths added by G_{m_x} to $G_{\mathcal{A}}$ for trip m, and with $P^m_{m_x \oplus \mathcal{A}}$ the set of all paths for trip m in $G_{m_x \oplus \mathcal{A}}$. Note that $P^m_{m_x|\mathcal{A}}$ is defined as $P^m_{m_x|\mathcal{A}} = P^m_{m_x \oplus \mathcal{A}} \setminus P^m_{\mathcal{A}}$. From this definition we obtain that $P^m_{m_x \oplus \mathcal{A}} = P^m_{m_x|\mathcal{A}} \cup P^m_{\mathcal{A}}$, with $P^m_{m_x|\mathcal{A}} \cap P^m_{\mathcal{A}} = \varnothing$. We also define $P_{\mathcal{A}} = \bigcup_{m \in \mathcal{M}} P^m_{\mathcal{A}}$.

Let us look at another example for this notation. Suppose that there exists only one trip $\mathcal{M} = \{(s^1, t^1, d^1)\}$. In Figure 4, we can see a trip spanning tree $G_{\mathcal{I}}$ for trip 1 (shown in black) and a trip path graph G_{1_1} for the same trip (shown in orange). Therefore, we will have $|P^1_{\mathcal{I}}| = |P^1_{1_1}| = 1$. These two sets will contain the black and the orange path respectively. The set $P^1_{1_1 \oplus \mathcal{I}} = P^1_{1_1|\mathcal{I}} \cup P^1_{\mathcal{I}}$ will contain four paths: three of which come from the set $P^1_{1_1|\mathcal{I}}$, which contains the path in $P^1_{1_1}$ $((s^1, w, r, v, g, h, t^1))$ and two new induced paths $((s^1, v, g, h, t^1), (s^1, w, r, v, t^1))$, and the last being the original path present in $P^1_{\mathcal{I}}$ $((s^1, v, t^1))$.

7. Properties of path additions

Having defined the problem of path additions, we turn our attention to its properties and, in particular to monotonicity and supermodularity.

Let us start by introducing the functions we will analyse. Given a graph template $G_{\mathcal{T}}$ with constant edge costs, a set of trips $\mathcal{M}$, and a trip spanning

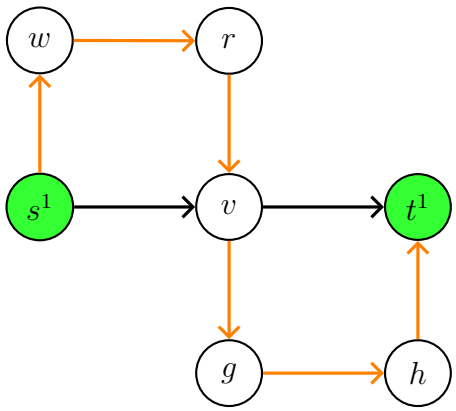


Figure 4: Illustration of the addition of a trip path graph. In this figure we have a trip spanning tree $G_{\mathcal{I}}$ for trip 1 (shown in black) to which we add a trip path graph G_{1_1} for the same trip (shown in orange). We see how, by only adding one trip path graph to the trip spanning tree, we obtain four total paths for trip 1, of which: one is the original one present in the trip spanning tree ((s^1, v, t^1)), one is present in G_{1_1} ($(s^1, w, r, v, g, h, t^1)$), and the remaining two ($(s^1, v, g, h, t^1), (s^1, w, r, v, t^1)$) are created from the addition.

tree $G_{\mathcal{I}}$, we denote as $\mathcal{G}$ the finite set of all possible trip path graphs. Note that $\mathcal{G} = \bigcup_{m \in \mathcal{M}} \mathcal{G}_m$ is the union of the disjoint sets of trip path graphs for each trip, where $\mathcal{G}_m = \{G_{m_1}, G_{m_2}, \dots, G_{m_x}, \dots\}$ is the set of all trip path graphs for trip m. We then define $\Lambda_{\text{MC}} : 2^{\mathcal{G}} \mapsto \mathbb{R}_{>0}$ which takes as input a subset of $\mathcal{G}$, adds it to $G_{\mathcal{I}}$, and outputs the cost of the MC routing on the resulting graph. This function can be written as follows:

$$\Lambda_{\text{MC}}(\mathcal{A}) = \sum_{p \in P_{\mathcal{A} \oplus \mathcal{I}}} x_p^{\text{MC}} c_p,$$

where $\mathbf{x}^{\text{MC}}$ is the solution to the MC problem on graph $G_{\mathcal{A} \oplus \mathcal{I}}$. $\mathcal{A} \subseteq \mathcal{G}$ is a set of trip path graphs, equally identified by the union of all the graphs in it, which we denote as $G_{\mathcal{A}} = \bigcup_{G_x \in \mathcal{A}} G_x$, containing the set of paths $P_{\mathcal{A}}$. The trip spanning tree $G_{\mathcal{I}}$ guarantees that when Λ_{MC} is computed on the empty set ($\Lambda_{\text{MC}}(\varnothing)$) there is still a feasible flow for each trip and each trip's source is connected to its destination.

Similarly, we define Λ_{SO} and Λ_{UE}, which operate on graphs with flow-dependant cost functions. We call $\mathcal{H}$ the set of all possible trip path graphs drawn from a template with flow-dependant costs. $\Lambda_{\text{SO}}, \Lambda_{\text{UE}} : 2^{\mathcal{H}} \mapsto \mathbb{R}_{>0}$ are

defined as follows:

$$\Lambda_{\mathrm{SO}}(\mathcal{A}) = \sum_{p \in P_{\mathcal{A} \oplus \mathcal{I}}} x_p^{\mathrm{SO}} c_p(x_p^{\mathrm{SO}})$$

$$\Lambda_{\mathrm{UE}}(\mathcal{A}) = \sum_{p \in P_{\mathcal{A} \oplus \mathcal{I}}} x_p^{\mathrm{UE}} c_p(x_p^{\mathrm{UE}}),$$

where $\mathbf{x}^{\mathrm{SO}}$ and $\mathbf{x}^{\mathrm{UE}}$ are the solutions to the SO and UE flow problems on graph $G_{\mathcal{A} \oplus \mathcal{I}}$.

It is important to understand that Λ_{MC} could be a particular case of Λ_{SO} with constant costs. The only difference is that Λ_{MC} is subject to the capacity constraint (1b) of the MC problem. While this constraint is ensured to be satisfied for the trip spanning tree (Property 4), we do not enforce the same for path additions.

We can use the functions just introduced to define NDPs via path additions. The problems will consist in minimizing one of the three functions introduced (depending on the routing formulation), subject to cardinality constraints, construction budgets, or general constraints formulated as matroids. The lower level of these NDPs is encapsulated in our objective functions.

To understand whether we can design near-optimal approximation algorithms for these NDPs, we are now interested in analysing the properties of the functions just introduced.

7.1. Monotonicity

Let us analyse the monotonicity of $\Lambda_{\mathrm{MC}}, \Lambda_{\mathrm{SO}}, \Lambda_{\mathrm{UE}}$.

Theorem 1 ([13])**.** Λ_{UE} *is* ***not*** *a monotonic non-increasing set function of the set* $\mathcal{H}$*.*

Proof. The proof is provided by the Braess' paradox [13]. His counterexample is discussed in Section 3.1. ■

Theorem 2. Λ_{SO} *is a monotonic non-increasing set function of the set* $\mathcal{H}$*.*

Proof. The proof can be found in Appendix B.1. ■

Theorem 3. Λ_{MC} *is a monotonic non-increasing set function of the set* $\mathcal{G}$*.*

Proof. The proof can be found in Appendix B.2. ■

The intuition behind the proofs of monotonicity is simple: when the objective is to minimise the social cost, like in MC and SO, the addition of a path can either provide a decrease in total cost or be ignored[2] by the agents. This is not the case for UE, as shown by Braess.

7.2. Supermodularity

The property of supermodularity of path additions can be informally explained through the following statement: 'adding a path to a transport network always grants greater or equal benefit to users than adding it to a superset of that network'. Although this property may appear intuitive at first, we show that it does not hold in general, even in the case where all the agents are cooperating to minimise the social cost. As in Braess' work, we present a counterexample to support our proofs, shown in Fig. 5 and 6.

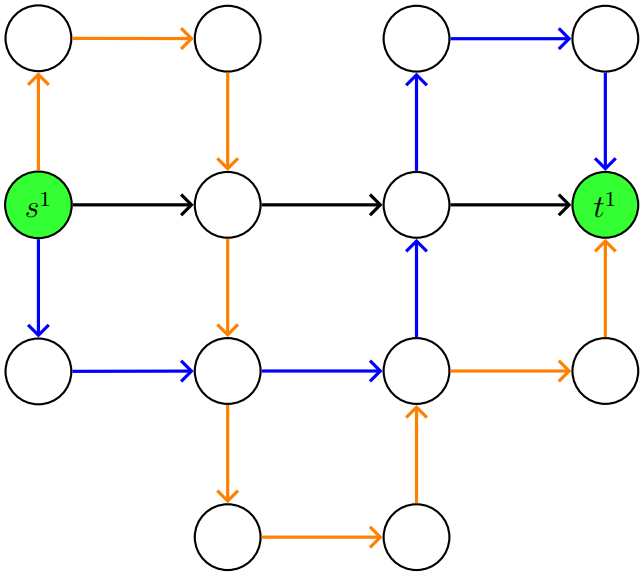


Figure 5: Network used as counterexample to the supermodularity of path additions to transport networks (Theorems 4, 5). In this example we have a trip spanning tree (shown in black) and two trip path graphs (shown in blue and orange). Fig.6 explains this counterexample step by step.

[2]Inconvenient additions can be ignored thanks to the fact that the trip spanning tree always grants a feasible starting solution.

If one can prove that a function $\Lambda : 2^{\mathcal{C}} \mapsto \mathbb{R}_{>0}$ is not supermodular on a set $\mathcal{C}' \subseteq \mathcal{C}$, it follows that Λ is not supermodular on $\mathcal{C}$. Therefore, to increase the generality of our proofs, we restrict the set of possible path additions by considering only trip path graphs that do not share any edges among them and with the trip spanning tree and have the same cost function and parameters for each edge. The sets $\mathcal{G}$ and $\mathcal{H}$ are thus restricted to

$$\begin{aligned}\mathcal{G}' = \{G_x | G_x \in \mathcal{G} \wedge \\ \forall G_y \in \mathcal{G}' \, (G_y \neq G_x \rightarrow \mathcal{E}_x \cap \mathcal{E}_y \cap \mathcal{E}_{\mathcal{I}} = \varnothing \wedge \\ \forall (i,j) \in \mathcal{E}_x, \forall (k,z) \in \mathcal{E}_y \, (u_{ij} = u_{kz} \wedge c_{ij} = c_{kz}))\}\end{aligned}$$

$$\begin{aligned}\mathcal{H}' = \{G_x | G_x \in \mathcal{H} \wedge \\ \forall G_y \in \mathcal{H}' \, (G_y \neq G_x \rightarrow \mathcal{E}_x \cap \mathcal{E}_y \cap \mathcal{E}_{\mathcal{I}} = \varnothing \wedge \\ \forall (i,j) \in \mathcal{E}_x, \forall (k,z) \in \mathcal{E}_y, \forall t \in \mathbb{R}_{\geqslant 0} \, (c_{ij}(t) = c_{kz}(t)))\}.\end{aligned}$$

Theorem 4. Λ_{MC} *is* ***not*** *a supermodular set function of the set the set* $\mathcal{G}'$*.*

Proof. We present a proof by counterexample. It is show that this is valid also in the case of only one trip. Thus, it is sufficient to show that we can find $\{x\}, \mathcal{Y} \subseteq \mathcal{G}'$ such that

$$\Lambda_{\text{MC}}(\varnothing) - \Lambda_{\text{MC}}(\{x\}) < \Lambda_{\text{MC}}(\mathcal{Y}) - \Lambda_{\text{MC}}(\mathcal{Y} \cup \{x\}). \tag{9}$$

Let us consider the graph depicted in Fig. 5. This transport network is composed by: the trip spanning tree $G_{\mathcal{I}}$, shown in black, the trip path graph $\{G_{1_y}\} = \mathcal{Y}$, shown in blue, and the trip path graph $G_{1_x} = x$, shown in orange.

The network has the following parameters:

- There is only one trip $\mathcal{M} = \{(s^1, t^1, 1)\}$
- Every edge has a capacity $\geqslant 1$
- Edges in the trip spanning tree have a cost of 3
- All other edges have a cost of 1.

$\Lambda_{\text{MC}}(\varnothing) = 9$ since all the agents are forced on the only available path (provided by $G_{\mathcal{I}}$). $\Lambda_{\text{MC}}(\{x\}) = 9$ since the addition provides three new paths, all with the same cost of 9. $\Lambda_{\text{MC}}(\mathcal{Y}) = 7$ since now the blue path has the

best cost and it can sustain all the flow. $\Lambda_{\text{MC}}(\mathcal{Y} \cup \{x\}) = 5$ since the blue and orange paths compose to form a solution with cost 5.

Therefore, Equation (9) holds. The counterexample is furtherly illustrated in Fig. 6. ■

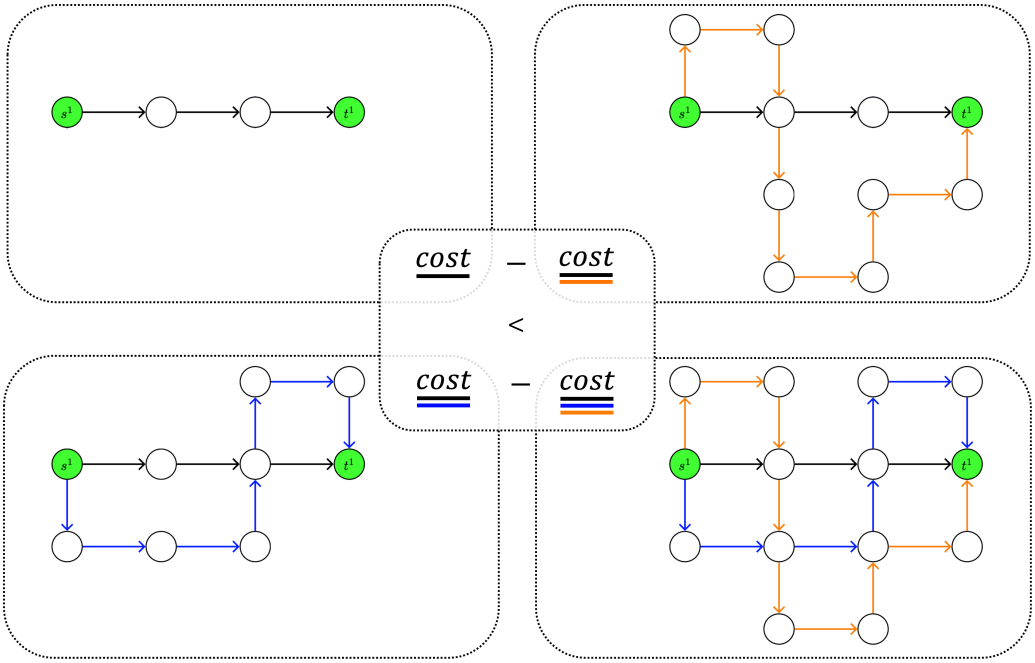


Figure 6: Visualisation of our counterexample (Theorems 4, 5) illustrating the non-supermodularity of path additions to transport networks. The cost represents the total travel time. In this scenario, the utility (shown as cost difference) of adding the orange path to the black-and-blue network *is greater* than the utility of adding the orange path just to the black subnetwork. This shows that the utility is not diminishing and, thus, the total travel time of coordinating agents is not supermodular with respect to path additions.

Theorem 5. Λ_{SO} *and* Λ_{UE} *are* ***not*** *supermodular set functions of the set the set* $\mathcal{H}'$*.*

Proof. The counterexample and the network structure are the same as in the proof of Theorem 4. The parameters are the following:

- There is only one trip $\mathcal{M} = \{(s^1, t^1, 5)\}$

- Every edge has a capacity $u_{ij} = 10$
- Edges have a length $l_{ij} = 1$ and a speed limit $v_{ij}^{\max} = 1$
- The Greenshields affine velocity function (Sec. 5.2.1) is used as the edge cost function.

To compute the Λ_{SO} and Λ_{UE} values of Equation (9), we use the Frank-Wolfe convex optimisation algorithm[3] [65]. We confirm the obtained values by exact hand calculations. When solving Equation (9) for Λ_{SO} we now obtain:

$$\Lambda_{\mathrm{SO}}(\varnothing) - \Lambda_{\mathrm{SO}}(\{x\}) < \Lambda_{\mathrm{SO}}(\mathcal{Y}) - \Lambda_{\mathrm{SO}}(\mathcal{Y} \cup \{x\})$$
$$30 - 29.28 < 27.47 - 24.97.$$

While for Λ_{UE} we obtain:

$$\Lambda_{\mathrm{UE}}(\varnothing) - \Lambda_{\mathrm{UE}}(\{x\}) < \Lambda_{\mathrm{UE}}(\mathcal{Y}) - \Lambda_{\mathrm{UE}}(\mathcal{Y} \cup \{x\})$$
$$30 - 30 < 30 - 26.66.$$

Both equations hold.

The same result can be shown with the BPR cost function (Sec. 5.2.2) for various choices of α_{ij} and β_{ij}. ■

While the non-supermodularity of Λ_{UE} was foreseeable given its non-monotonicity, the non-supermodularity of path additions for cooperative routing (Λ_{MC} and Λ_{SO}) characterises an important insight for transport network designers.

7.3. Supermodularity of parallel paths

Having proven the non-supermodularity of our functions on $\mathcal{G}'$ and $\mathcal{H}'$, we are now concerned with the following question: *is there a subset of the space of possible path additions where we can prove supermodularity?* The answer to this question is positive. We will now illustrate such subsets and the relative supermodularity proofs for MC (Sec. 7.3.1) and for SO and UE (Sec. 7.3.2).

[3] `https://github.com/matteobettini/Traffic-Assignment-Frank-Wolfe-2021`

7.3.1. MC

To prove supermodularity in the MC problem, we now consider the case of parallel trip path graphs. In particular, we also restrict our attention to the single trip case, as, due to the parallel nature of the paths we consider, each trip can be treated independently.

Thus, given a trip (s, t, d), we restrict the space of possible trip path graph additions to

$$\mathcal{G}'' = \{G_x | G_x \in \mathcal{G} \wedge \forall (i,j) \in \mathcal{E}_x (u_{ij} \geqslant d) \ \wedge \forall G_y \in \mathcal{G}'' \\ (G_y \neq G_x \rightarrow \mathcal{E}_x \cap \mathcal{E}_y \cap \mathcal{E}_{\mathcal{I}} = \varnothing \ \wedge \mathcal{V}_x \cap \mathcal{V}_y \cap \mathcal{V}_{\mathcal{I}} = \{s, t\})\},$$

which means that we only consider parallel trip path graphs with different costs which can individually hold all the demand.

The graphs considered in this section are obtained after multiple additions (taken from $\mathcal{G}''$) to a trip spanning tree. They will be represented using two nodes: origin (s) and destination (t), and a set of directed edges, representing the parallel paths connecting s with t. Each path has a cost $c_p = \sum_{(i,j)\in p} c_{ij}$ and a capacity $u_p = \min_{(i,j)\in p} u_{ij}$. An example of such network can be seen in Figure 7.

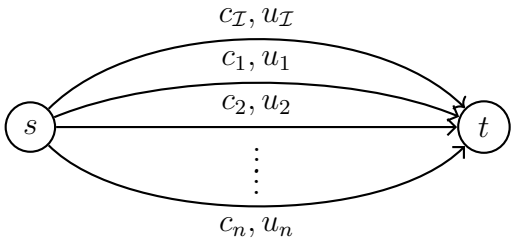


Figure 7: An example of a graph in the single trip parallel paths case. Here we can see the trip spanning tree $G_{\mathcal{I}}$ (which in this scenario is always consisting of just one path with cost $c_{\mathcal{I}}$ and capacity $u_{\mathcal{I}}$) and some trip path graphs numbered from 1 to n. The notation on the edges shows cost and capacity of each path, separated by a comma.

In this scenario, the MC assignment becomes trivial, as the best solution is always to route the whole flow demand on the minimum cost path available in the graph. Hence, we can rewrite Λ_{MC} as follows:

$$\Lambda_{\text{MC}}(\mathcal{A}) = \sum_{p \in P_{\mathcal{A}\oplus\mathcal{I}}} x_p^{\text{MC}} c_p = d \min \{c_p | p \in P_{\mathcal{A}\oplus\mathcal{I}}\}. \tag{10}$$

Lemma 5. *Given the sets of paths $P_{\mathcal{A}}$ and $P_{\mathcal{B}}$ obtained respectively from graphs $G_{\mathcal{A}}, G_{\mathcal{B}}$, we have that $P_{\mathcal{A}}, P_{\mathcal{B}} \subseteq P_{\mathcal{A}\oplus\mathcal{B}}$.*

Proof. The proof follows from the fact that graph $G_{\mathcal{A}\oplus\mathcal{B}}$ is defined as $G_{\mathcal{A}\oplus\mathcal{B}} = G_{\mathcal{A}} \cup G_{\mathcal{B}}$ and thus we can write $P_{\mathcal{A}} \cup P_{\mathcal{B}} \subseteq P_{\mathcal{A}\oplus\mathcal{B}}$. Hence, we can rewrite the thesis of Lemma 5 as $P_{\mathcal{A}}, P_{\mathcal{B}} \subseteq P_{\mathcal{A}} \cup P_{\mathcal{B}}$ which is true by the definition of the union over sets. ■

Theorem 6. *Λ_{MC} is a supermodular set function of the set the set $\mathcal{G}''$.*

Proof. The proof can be found in Appendix B.3. ■

The proof depends on the fact that each path can sustain the entire flow. Therefore, the addition of a new path can either:

- Make no change (because it has a higher cost than the best path already in the graph) and will continue to make no change when considering adding it to a superset of the graph.
- Make an improvement (because it is the new best path). The improvement is its cost minus the previous best cost. Adding this path to superset of the graph will at best make the same improvement (if the superset graph has the same best cost) or a smaller improvement (if the superset graph has a better best cost than previously).

7.3.2. SO and UE

To prove supermodularity in the SO and UE problem, we consider the case of parallel identical trip path graphs. In particular, we restrict our attention to the single trip case, as, due to the parallel nature of the paths we consider, each trip can be treated independently. The set of possible path additions is $\mathcal{H}'' \subseteq \mathcal{H}$, defined as

$$\begin{aligned}\mathcal{H}'' = \{G_x | G_x \in \mathcal{H} \ \wedge \forall G_y \in \mathcal{H}'' \, (G_y \neq G_x \rightarrow \\ \mathcal{E}_x \cap \mathcal{E}_y \cap \mathcal{E}_{\mathcal{I}} = \varnothing \ \wedge \mathcal{V}_x \cap \mathcal{V}_y \cap \mathcal{V}_{\mathcal{I}} = \{s,t\} \wedge \\ \forall p \in P_x, \forall k \in P_y, \forall t \in \mathbb{R}_{\geqslant 0} \, (c_p(t) = c_k(t))\}.\end{aligned}$$

All the paths being identical, we can treat them as edges with the same length l, the same maximum velocity $v_{\max}$ and the same capacity u. Thanks to these assumptions, we can use the following flow assignment:

$$x_p = \frac{d}{n_P} \quad \forall p \in P, \tag{11}$$

where $n_P = |P|$ is the number of parallel paths available. We show that this assignment is optimal for both SO and UE on $\mathcal{H}''$ when using Greenshields' cost function.

Lemma 6. *Given an instance* $(G_{\mathcal{A}\oplus\mathcal{I}}, \mathcal{M}, \mathbf{c})$ *where* $\mathcal{A} \subseteq \mathcal{H}''$, $\mathcal{M} = \{(s,t,d)\}$, *and* $\mathbf{c}$ *is a vector of Greenshields affine velocity functions, assignment* (11) *is an optimal solution for the UE problem.*

Proof. By Lemma 2 we know that if we can show that $\forall m \in \mathcal{M}$ and $p_1, p_2 \in P^m_{\mathcal{A}\oplus\mathcal{I}}$ with $x_{p_1} > 0$, we have $c_{p_1}(x_{p_1}) \leqslant c_{p_2}(x_{p_2})$ then the assignment is at UE.

Given that all paths are identical and, under assignment (11), all flows are positive, we can rewrite

$$c_{p_1}(x_{p_1}) \leqslant c_{p_2}(x_{p_2})$$

as

$$\frac{l}{v_{\max}\left(1 - \frac{d}{|P_{\mathcal{A}\oplus\mathcal{I}}|u}\right)} \leqslant \frac{l}{v_{\max}\left(1 - \frac{d}{|P_{\mathcal{A}\oplus\mathcal{I}}|u}\right)},$$

which is always true. ■

Lemma 7. *Given an instance* $(G_{\mathcal{A}\oplus\mathcal{I}}, \mathcal{M}, \mathbf{c})$ *where* $\mathcal{A} \subseteq \mathcal{H}''$ *and* $\mathcal{M} = \{(s,t,d)\}$, *and* $\mathbf{c}$ *is a vector of Greenshields affine velocity functions, assignment* (11) *is an optimal solution for the SO problem.*

Proof. By Lemma 1 we know that if we can show that $\forall m \in \mathcal{M}$ and $p_1, p_2 \in P^m_{\mathcal{A}\oplus\mathcal{I}}$ with $x_{p_1} > 0$, we have $k'_{p_1}(x_{p_1}) \leqslant k'_{p_2}(x_{p_2})$ than the assignment is at UE.

Given that all paths are identical and, under assignment (11), all flows are positive, we can rewrite

$$k'_{p_1}(x_{p_1}) \leqslant k'_{p_2}(x_{p_2})$$

as

$$\left(\frac{d}{|P_{\mathcal{A}\oplus\mathcal{I}}|} \cdot \frac{l}{v_{\max}\left(1 - \frac{d}{|P_{\mathcal{A}\oplus\mathcal{I}}|u}\right)}\right)' \leqslant \left(\frac{d}{|P_{\mathcal{A}\oplus\mathcal{I}}|} \cdot \frac{l}{v_{\max}\left(1 - \frac{d}{|P_{\mathcal{A}\oplus\mathcal{I}}|u}\right)}\right)',$$

which is always true. ■

Theorem 7. *Λ_{SO} and Λ_{UE} are supermodular set functions of the set $\mathcal{H}''$.*

Proof. The proof can be found in Appendix B.4. ■

The proof leverages the concept that, after every new addition, the flow will split in equal parts on the available paths. Thus, the benefit of an addition when there are fewer paths available is intuitively greater than the benefit of such addition in a network with more paths.

8. Conclusion

To conclude, we have analysed and proven several fundamental properties of path additions to transport networks. We have done this for the most popular routing formulations in traffic research. It is shown that, while the total travel time of cooperative agents (MC and SO) is monotonic non-increasing with respect to path additions, it is not supermodular. This impossibility result constitutes a fundamental insight for transport network designers as it informs that approximation algorithms for this problem cannot be designed by leveraging supermodularity in the general case.

On the other hand, we have introduced special cases that consider parallel path additions, where supermodularity holds. These instances could map to real-world scenarios where parallel paths are guaranteed by using dedicated lanes that are either physically or virtually enforced.

Our investigation and categorisation of network design properties under different routing paradigms was motivated by the prospective impact of CAVs on current transport networks. It is conceivable that the widespread adoption of CAVs could lead to a redesign of our mobility infrastructure. More work is needed in this area to understand how network design solutions can be devised and applied to the problem of designing future mobility systems and we believe that the present work constitutes a step towards this overarching goal.

Declaration of competing interest

none

Acknowledgement

We gratefully acknowledge the support of ERC Project 949940 (gAIa) and of ARL grant DCIST CRA W911NF-17-2-0181.

Appendix A. Notation used in the paper

G_x	Graph x
$\mathcal{V}_x$	Node set of graph x
$\mathcal{E}_x$	Edge set of graph x
P_x	Path set of graph x
P_x^m	Path set of graph x w.r.t. trip m
$P_{y\|x}^m$	Paths added to graph x by the addition of graph y w.r.t. trip m
$m \in \mathcal{M}$	Trip m in the set of all trips
s^m	Source of trip m
t^m	Destination of trip m
d^m	Demand of trip m
d_i^m	Demand of trip m at node i
c_{ij}	Cost of edge (i, j)
c_p	Cost of path p
u_{ij}	Capacity of edge (i, j)
u_p	Capacity of path p
l_{ij}	Length of edge (i, j)
$v_{ij}^{\max}$	Maximum velocity on edge (i, j)
x_{ij}	Flow on edge (i, j)
x_{ij}^m	Flow on edge (i, j) w.r.t. trip m
x_p	Flow on path p
$\mathbf{x}$	Vector of all path flows
$G_{\mathcal{I}}$	Trip spanning tree
$G_{\mathcal{T}}$	Template graph
G_{m_x}	Trip path graph x w.r.t. trip m
$\mathcal{G}$	Set of all path additions with constant cost
$\mathcal{H}$	Set of all path additions with flow-dependent cost
$\mathcal{G}'$	Subset of $\mathcal{G}$, where paths do not share edges and all edges have the same cost and capacity
$\mathcal{H}'$	Subset of $\mathcal{H}$, where paths do not share edges and all edges have the same properties
$\mathcal{G}''$	Subset of $\mathcal{G}'$, where paths do not share nodes and can sustain the whole flow demand
$\mathcal{H}''$	Subset of $\mathcal{H}'$, where paths do not share nodes and have the same properties
Λ_{MC}	Objective function that takes a graph as input and computes the total travel time using MC routing
Λ_{SO}	Objective function that takes a graph as input and computes the total travel time using SO routing
Λ_{UE}	Objective function that takes a graph as input and computes the total travel time using UE routing

Appendix B. Proofs

Appendix B.1. Proof of Theorem 2

Theorem 2. Λ_{SO} *is a monotonic non-increasing set function of the set* $\mathcal{H}$*.*

Proof. We need to show that given two sets of trip path graphs $\mathcal{A}$ and $\mathcal{B}$ such that $\mathcal{A} \subseteq \mathcal{B} \subseteq \mathcal{H}$, we have that $\Lambda(\mathcal{B}) \leqslant \Lambda(\mathcal{A})$.

We can rewrite $\Lambda_{\mathrm{SO}}(\mathcal{A})$ as

$$\sum_{p \in P_{\mathcal{A} \oplus \mathcal{I}}} x_p^{\mathrm{SO}} c_p(x_p^{\mathrm{SO}}) = \min_{\mathbf{x}} \left\{ \sum_{p \in P_{\mathcal{A} \oplus \mathcal{I}}} x_p c_p(x_p) \right\},$$

subject to constraints (5b) and (5c). $\Lambda(\mathcal{B}) \leqslant \Lambda(\mathcal{A})$ thus becomes

$$\min_{\mathbf{x}} \left\{ \sum_{p \in P_{\mathcal{B} \oplus \mathcal{I}}} x_p c_p(x_p) \right\} \leqslant \min_{\mathbf{x}} \left\{ \sum_{p \in P_{\mathcal{A} \oplus \mathcal{I}}} x_p c_p(x_p) \right\}.$$

If $\mathcal{A} \subseteq \mathcal{B}$ there exists $\mathcal{Y}$ s.t. $\mathcal{B} = \mathcal{A} \cup \mathcal{Y}$. Therefore we can rewrite the above equation as

$$\min_{\mathbf{x}} \left\{ \sum_{p \in P_{\mathcal{Y} \oplus \mathcal{A} \oplus \mathcal{I}}} x_p c_p(x_p) \right\} \leqslant \min_{\mathbf{x}} \left\{ \sum_{p \in P_{\mathcal{A} \oplus \mathcal{I}}} x_p c_p(x_p) \right\}.$$

We know that $P_{\mathcal{Y} \oplus \mathcal{A} \oplus \mathcal{I}}$ can be decomposed into its two disjoint components $P_{\mathcal{A} \oplus \mathcal{I}}$ and $P_{\mathcal{Y} | \mathcal{A} \oplus \mathcal{I}}$. Thus we can rewrite

$$\min_{\mathbf{x}} \left\{ \sum_{p \in P_{\mathcal{A} \oplus \mathcal{I}}} x_p c_p(x_p) + \sum_{p \in P_{\mathcal{Y} | \mathcal{A} \oplus \mathcal{I}}} x_p c_p(x_p) \right\} \leqslant \min_{\mathbf{x}} \left\{ \sum_{p \in P_{\mathcal{A} \oplus \mathcal{I}}} x_p c_p(x_p) \right\}.$$

If we turn our attention to the left side of the inequality and particularly to the second term of the sum, we can set $x_p = 0,\ \forall p \in P_{\mathcal{Y} | \mathcal{A} \oplus \mathcal{I}}$, thus obtaining, with this additional constraint

$$\min_{\mathbf{x}} \left\{ \sum_{p \in P_{\mathcal{A} \oplus \mathcal{I}}} x_p c_p(x_p) \right\} = \min_{\mathbf{x}} \left\{ \sum_{p \in P_{\mathcal{A} \oplus \mathcal{I}}} x_p c_p(x_p) \right\}. \tag{B.1}$$

This confirms our theorem since the minimum of a function subject to a constraint is greater or equal to the minimum of the function without said constraint. ■

Appendix B.2. Proof of Theorem 3

Theorem 3. Λ_{MC} *is a monotonic non-increasing set function of the set* $\mathcal{G}$*.*

Proof. The proof follows the same procedure as in Theorem 2, in the case of constant costs. Property (4) of a trip spanning tree guarantees that when we impose the additional constraint leading to Equation (B.1), we do not break constraint (1b) of MC routing and the problem remains feasible. ■

Appendix B.3. Proof of Theorem 6

Theorem 6. Λ_{MC} *is a supermodular set function of the set the set* $\mathcal{G}''$*.*

Proof. We need to show that, given two subsets $\mathcal{A}$ and $\mathcal{B}$ such that $\mathcal{A} \subseteq \mathcal{B} \subseteq \mathcal{G}''$ and $x \in \mathcal{G}'' \setminus \mathcal{B}$, it holds that

$$\Lambda(\mathcal{A}) - \Lambda(\mathcal{A} \cup \{x\}) \geqslant \Lambda(\mathcal{B}) - \Lambda(\mathcal{B} \cup \{x\}).$$

Knowing that $\mathcal{A} \subseteq \mathcal{B}$ we can define $\mathcal{B} = \mathcal{Y} \cup \mathcal{A}$ with $\mathcal{A} \cap \mathcal{Y} = \varnothing$. This yields

$$\Lambda(\mathcal{A}) - \Lambda(\mathcal{A} \cup \{x\}) \geqslant \Lambda(\mathcal{Y} \cup \mathcal{A}) - \Lambda(\mathcal{Y} \cup \mathcal{A} \cup \{x\}),$$

which, according to Equation (10), we can rewrite as

$$\begin{aligned} d \min \{c_p | p \in P_{\mathcal{A} \oplus \mathcal{I}}\} - d \min \{c_p | p \in P_{x \oplus \mathcal{A} \oplus \mathcal{I}}\} \geqslant \\ d \min \{c_p | p \in P_{\mathcal{Y} \oplus \mathcal{A} \oplus \mathcal{I}}\} - d \min \{c_p | p \in P_{x \oplus \mathcal{Y} \oplus \mathcal{A} \oplus \mathcal{I}}\}. \end{aligned} \tag{B.2}$$

The term d can be simplified knowing that $d \geqslant 0$. We proceed to consider the following two cases:

Case 1: $\min \{c_p | p \in P_{x \oplus \mathcal{A} \oplus \mathcal{I}}\} \geqslant \min \{c_p | p \in P_{\mathcal{Y} \oplus \mathcal{A} \oplus \mathcal{I}}\}$. This implies that

$$\min \{c_p | p \in P_{\mathcal{Y} \oplus \mathcal{A} \oplus \mathcal{I}}\} = \min \{c_p | p \in P_{x \oplus \mathcal{Y} \oplus \mathcal{A} \oplus \mathcal{I}}\},$$

Equation (B.2) becomes

$$\min\{c_p | p \in P_{\mathcal{A}\oplus\mathcal{I}}\} - \min\{c_p | p \in P_{x\oplus\mathcal{A}\oplus\mathcal{I}}\} \geqslant 0. \tag{B.3}$$

By Lemma 5 we know that $P_{\mathcal{A}\oplus\mathcal{I}} \subseteq P_{x\oplus\mathcal{A}\oplus\mathcal{I}}$. Recalling that the minimum of a set is always greater or equal than the minimum of one of its supersets, (B.3) holds.

Case 2: $\min\{c_p | p \in P_{x\oplus\mathcal{A}\oplus\mathcal{I}}\} < \min\{c_p | p \in P_{\mathcal{Y}\oplus\mathcal{A}\oplus\mathcal{I}}\}$. This implies that

$$\min\{c_p | p \in P_{x\oplus\mathcal{A}\oplus\mathcal{I}}\} = \min\{c_p | p \in P_{x\oplus\mathcal{Y}\oplus\mathcal{A}\oplus\mathcal{I}}\},$$

Equation (B.2) becomes

$$\min\{c_p | p \in P_{\mathcal{A}\oplus\mathcal{I}}\} \geqslant \min\{c_p | p \in P_{\mathcal{Y}\oplus\mathcal{A}\oplus\mathcal{I}}\}. \tag{B.4}$$

By Lemma 5 we know that $P_{\mathcal{A}\oplus\mathcal{I}} \subseteq P_{\mathcal{Y}\oplus\mathcal{A}\oplus\mathcal{I}}$. Recalling that the minimum of a set is always greater or equal than the minimum of one of its supersets, (B.4) holds. ■

Appendix B.4. Proof of Theorem 7

Theorem 7. Λ_{SO} *and* Λ_{UE} *are supermodular set functions of the set* $\mathcal{H}''$.

Proof. As we have shown with Lemma 6 and Lemma 7 that Λ_{UE} and Λ_{SO} are equivalent on $\mathcal{H}''$, we proceed to prove the theorem for Λ_{SO}, the result will be also valid for Λ_{UE}. As before, we need to prove that, given two subsets $\mathcal{A}$ and $\mathcal{B}$ such that $\mathcal{A} \subseteq \mathcal{B} \subseteq \mathcal{H}''$ and $x \in \mathcal{H}'' \setminus \mathcal{B}$, it holds that

$$\Lambda_{\text{SO}}(\mathcal{A}) - \Lambda_{\text{SO}}(\mathcal{A} \cup \{x\}) \geqslant \Lambda_{\text{SO}}(\mathcal{B}) - \Lambda_{\text{SO}}(\mathcal{B} \cup \{x\}).$$

Knowing that $\mathcal{A} \subseteq \mathcal{B}$ we can define $\mathcal{B} = \mathcal{Y} \cup \mathcal{A}$ with $\mathcal{A} \cap \mathcal{Y} = \varnothing$. This yields

$$\Lambda_{\text{SO}}(\mathcal{A}) - \Lambda_{\text{SO}}(\mathcal{A} \cup \{x\}) \geqslant \Lambda_{\text{SO}}(\mathcal{Y} \cup \mathcal{A}) - \Lambda_{\text{SO}}(\mathcal{Y} \cup \mathcal{A} \cup \{x\}).$$

We know that the trip spanning tree $G_{\mathcal{I}}$ contains only one path $|P_{\mathcal{I}}| = 1$. We call the number of paths in $\mathcal{A}$ and $\mathcal{Y}$, $n_a \geqslant 0$ and $n_y \geqslant 0$ respectively. We also know that $|P_{\mathcal{A}\oplus\mathcal{I}}| = n_a + 1$. We fix $n = n_a + 1$.

By Lemma 7 we know that Equation (11) is an optimal assignment, thus, using the definition of Λ_{SO}, we can rewrite

$$dc_p\left(\frac{d}{n}\right) - dc_p\left(\frac{d}{n+1}\right) \geqslant dc_p\left(\frac{d}{n+n_y}\right) - dc_p\left(\frac{d}{n+n_y+1}\right).$$

By expanding using the Greenshields cost in Equation 3, we have

$$d\frac{l}{v^{\max}\left(1-\frac{d}{nu}\right)} - d\frac{l}{v^{\max}\left(1-\frac{d}{(n+1)u}\right)} \geqslant d\frac{l}{v^{\max}\left(1-\frac{d}{(n+n_y)u}\right)} - d\frac{l}{v^{\max}\left(1-\frac{d}{(n+n_y+1)u}\right)}.$$

We know that $d, l, v^{\max}, u \geqslant 0$. Thus, we can apply a series of rewrites:

$$\frac{1}{(nu-d)((n+1)u-d)} \geqslant \frac{1}{((n+n_y)u-d)((n+n_y+1)u-d)}.$$

Thanks to Property (4) of a trip spanning tree, we know that $u \geqslant d$, thus both denominators are non-negative and we have

$$((n+n_y)u-d)((n+n_y+1)u-d) \geqslant (nu-d)((n+1)u-d)$$

$$n_y u(n_y u + u + 2nu - 2d) \geqslant 0,$$

which, by the fact that $n \geqslant 1$ and $u \geqslant d$, is always true. ■

References

[1] M. Patriksson, The traffic assignment problem: models and methods, Courier Dover Publications, 2015.

[2] D. Bertsimas, S. S. Patterson, The air traffic flow management problem with enroute capacities, Operations research 46 (3) (1998) 406–422.

[3] N. Bellomo, C. Dogbe, On the modelling crowd dynamics from scaling to hyperbolic macroscopic models, Mathematical Models and Methods in Applied Sciences 18 (supp01) (2008) 1317–1345.

[4] G. R. Ash, Dynamic routing in telecommunications networks, McGraw-Hill Professional, 1997.

[5] S. A. Bagloee, M. Tavana, M. Asadi, T. Oliver, Autonomous vehicles: challenges, opportunities, and future implications for transportation policies, Journal of modern transportation 24 (4) (2016) 284–303.

[6] F. Duarte, C. Ratti, The impact of autonomous vehicles on cities: A review, Journal of Urban Technology 25 (4) (2018) 3–18.

[7] N. Hyldmar, Y. He, A. Prorok, A fleet of miniature cars for experiments in cooperative driving, in: 2019 International Conference on Robotics and Automation (ICRA), IEEE, 2019, pp. 3238–3244.

[8] W. Wolf, Car mania: a critical history of transport, Pluto Press, 1996.

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

[10] T. L. Magnanti, R. T. Wong, Network design and transportation planning: Models and algorithms, Transportation science 18 (1) (1984) 1–55.

[11] S. Fujishige, Submodular functions and optimization, Elsevier, 2005.

[12] A. Prorok, Robust assignment using redundant robots on transport networks with uncertain travel time, IEEE Transactions on Automation Science and Engineering 17 (4) (2020) 2025–2037.

[13] D. Braess, A. Nagurney, T. Wakolbinger, On a paradox of traffic planning, Transportation science 39 (4) (2005) 446–450.

[14] R. K. Ahuja, T. L. Magnanti, J. B. Orlin, Network flows, Cambridge, Mass.: Alfred P. Sloan School of Management, Massachusetts, 1988.

[15] T. Roughgarden, É. Tardos, How bad is selfish routing?, Journal of the ACM (JACM) 49 (2) (2002) 236–259.

[16] J. G. Wardrop, Road paper. some theoretical aspects of road traffic research., Proceedings of the institution of civil engineers 1 (3) (1952) 325–362.

[17] H. Yang, M. G. Bell, Models and algorithms for road network design: A review and some new developments, Transport Reviews 18 (3) (1998) 257–278. doi:10.1080/01441649808717016.

[18] A. Chen, Z. Zhou, P. Chootinan, S. Ryu, C. Yang, S. Wong, Transport network design problem under uncertainty: a review and new developments, Transport Reviews 31 (6) (2011) 743–768.

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

[20] M. Simaan, J. B. Cruz, On the stackelberg strategy in nonzero-sum games, Journal of Optimization Theory and Applications 11 (5) (1973) 533–555.

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

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

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

[24] P. Marcotte, Network optimization with continuous control parameters, Transportation Science 17 (2) (1983) 181–197.

[25] G. Ziyou, S. Yifan, A reserve capacity model of optimal signal control with user-equilibrium route choice, Transportation Research Part B: Methodological 36 (4) (2002) 313–323.

[26] R. Cole, Y. Dodis, T. Roughgarden, How much can taxes help selfish routing?, Journal of Computer and System Sciences 72 (3) (2006) 444–467.

[27] H. Yang, Sensitivity analysis for the elastic-demand network equilibrium problem with applications, Transportation Research Part B: Methodological 31 (1) (1997) 55–70.

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

[29] Z. Drezner, G. O. Wesolowsky, Network design: selection and design of links and facility location, Transportation Research Part A: Policy and Practice 37 (3) (2003) 241–256.

[30] H. Zhang, Z. Gao, Two-way road network design problem with variable lanes, Journal of Systems Science and Systems Engineering 16 (1) (2007) 50–61.

[31] Z. Drezner, S. Salhi, Using hybrid metaheuristics for the one-way and two-way network design problem, Naval Research Logistics (NRL) 49 (5) (2002) 449–463.

[32] H. Poorzahedy, M. A. Turnquist, Approximate algorithms for the discrete network design problem, Transportation Research Part B: Methodological 16 (1) (1982) 45–55.

[33] H. Poorzahedy, F. Abulghasemi, Application of ant system to network design problem, Transportation 32 (3) (2005) 251–273.

[34] C. F. Daganzo, Y. Sheffi, On stochastic models of traffic assignment, Transportation science 11 (3) (1977) 253–274.

[35] S. Peeta, A. K. Ziliaskopoulos, Foundations of dynamic traffic assignment: The past, the present and the future, Networks and spatial economics 1 (3-4) (2001) 233–265.

[36] K. Spieser, K. Treleaven, R. Zhang, E. Frazzoli, D. Morton, M. Pavone, Toward a systematic approach to the design and evaluation of automated mobility-on-demand systems: A case study in singapore, in: Road vehicle automation, Springer, 2014, pp. 229–245.

[37] O. Ben-Ayed, D. E. Boyce, C. E. Blair III, A general bilevel linear programming formulation of the network design problem, Transportation Research Part B: Methodological 22 (4) (1988) 311–318.

[38] G. B. Dantzig, R. P. Harvey, Z. F. Lansdowne, D. W. Robinson, S. F. Maier, Formulating and solving the network design problem by decomposition, Transportation Research Part B: Methodological 13 (1) (1979) 5–17.

[39] P. Marcotte, G. Savard, Bilevel programming: A combinatorial perspective, in: Graph theory and combinatorial optimization, Springer, 2005, pp. 191–217.

[40] Z.-Q. Luo, J.-S. Pang, D. Ralph, Mathematical programs with equilibrium constraints, Cambridge University Press, 1996.

[41] M. Chen, A. S. Alfa, A network design algorithm using a stochastic incremental traffic assignment approach, Transportation Science 25 (3) (1991) 215–224.

[42] H. Poorzahedy, O. M. Rouhani, Hybrid meta-heuristic algorithms for solving network design problem, European Journal of Operational Research 182 (2) (2007) 578–596.

[43] Y. Yin, Genetic-algorithms-based approach for bilevel programming models, Journal of transportation engineering 126 (2) (2000) 115–120.

[44] T. Xu, H. Wei, G. Hu, Study on continuous network design problem using simulated annealing and genetic algorithm, Expert Systems with Applications 36 (2) (2009) 1322–1328.

[45] T. V. Mathew, S. Sharma, Capacity expansion problem for large urban transportation networks, Journal of Transportation Engineering 135 (7) (2009) 406–415.

[46] S. A. Bagloee, M. Sarvi, M. Patriksson, A hybrid branch-and-bound and benders decomposition algorithm for the network design problem, Computer-Aided Civil and Infrastructure Engineering 32 (4) (2017) 319–343.

[47] E. B. Khalil, B. Dilkina, L. Song, Scalable diffusion-aware optimization of network topology, in: Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining, 2014, pp. 1226–1235.

[48] N. Mehr, R. Horowitz, A submodular approach for optimal sensor placement in traffic networks, in: 2018 Annual American Control Conference (ACC), IEEE, 2018, pp. 6353–6358.

[49] I. Shames, T. H. Summers, Rigid network design via submodular set function optimization, IEEE Transactions on Network Science and Engineering 2 (3) (2015) 84–96.

[50] A. Gupta, B. Dilkina, Budget-constrained demand-weighted network design for resilient infrastructure, in: 2019 IEEE 31st International Conference on Tools with Artificial Intelligence (ICTAI), IEEE, 2019, pp. 456–463.

[51] R. Steinberg, W. I. Zangwill, The prevalence of braess' paradox, Transportation Science 17 (3) (1983) 301–318.

[52] E. I. Pas, S. L. Principio, Braess' paradox: Some new insights, Transportation Research Part B: Methodological 31 (3) (1997) 265–276.

[53] G. Valiant, T. Roughgarden, Braess's paradox in large random graphs, Random Structures & Algorithms 37 (4) (2010) 495–515.

[54] D. M. Topkis, Supermodularity and complementarity, Princeton university press, 2011.

[55] L. R. Ford Jr, D. R. Fulkerson, Flows in networks, Vol. 54, Princeton university press, 2015.

[56] B. Greenshields, J. Bibbins, W. Channing, H. Miller, A study of traffic capacity, in: Highway research board proceedings, Vol. 1935, National Research Council (USA), Highway Research Board, 1935.

[57] F. Siebel, W. Mauser, On the fundamental diagram of traffic flow, SIAM Journal on Applied Mathematics 66 (4) (2006) 1150–1162.

[58] D. B. Work, S. Blandin, O.-P. Tossavainen, B. Piccoli, A. M. Bayen, A traffic model for velocity data assimilation, Applied Mathematics Research eXpress 2010 (1) (2010) 1–35.

[59] Y. Sheffi, Equilibrium analysis with mathematical programming methods, Massachusetts: Institute of Technology (1985).

[60] D. Branston, Link capacity functions: A review, Transportation research 10 (4) (1976) 223–236.

[61] A. L. Peressini, F. E. Sullivan, J. J. Uhl Jr, The mathematics of nonlinear programming, Springer-Verlag, 1988.

[62] M. Beckmann, C. B. McGuire, C. B. Winsten, Studies in the economics of transportation, Tech. rep. (1956).

[63] S. C. Dafermos, F. T. Sparrow, The traffic assignment problem for a general network, Journal of Research of the National Bureau of Standards B 73 (2) (1969) 91–118.

[64] M. Frank, P. Wolfe, et al., An algorithm for quadratic programming, Naval research logistics quarterly 3 (1-2) (1956) 95–110.

[65] L. J. LeBlanc, E. K. Morlok, W. P. Pierskalla, An efficient approach to solving the road network equilibrium traffic assignment problem, Transportation research 9 (5) (1975) 309–318.

[66] T. Roughgarden, Selfish routing and the price of anarchy, MIT press, 2005.

[67] T. Roughgarden, The price of anarchy is independent f the network topology, Journal of Computer and System Sciences 67 (2) (2003) 341–364.

[68] A. C. Pigou, N. Aslanbeigui, The economics of welfare, Routledge, 2017.