

Shortest Paths in Graphs of Convex Sets

Tobia Marcucci*, Jack Umenberger, Pablo A. Parrilo, and Russ Tedrake

Department of Electrical Engineering and Computer Science
Massachusetts Institute of Technology

Abstract

Given a graph, the shortest-path problem requires finding a sequence of edges with minimum cumulative length that connects a source vertex to a target vertex. We consider a generalization of this classical problem in which the position of each vertex in the graph is a continuous decision variable, constrained to lie in a corresponding convex set. The length of an edge is then defined as a convex function of the positions of the vertices it connects. Problems of this form arise naturally in motion planning of autonomous vehicles, robot navigation, and even optimal control of hybrid dynamical systems. The price for such a wide applicability is the complexity of this problem, which is easily seen to be NP-hard. Our main contribution is a strong mixed-integer convex formulation based on perspective functions. This formulation has a very tight convex relaxation and makes it possible to efficiently find globally-optimal paths in large graphs and in high-dimensional spaces.

1 Introduction

The Shortest-Path Problem (SPP) is one of the most deeply-studied problems in combinatorial optimization. In its single-source single-target version, this problem asks for a path of minimum length connecting two prescribed vertices of a graph, with the length of a path being defined as the sum of the length of its edges. Typically, the edge lengths are fixed scalars, given as problem data, and the assumptions made on their values have a dramatic impact on the problem complexity [82, Chapters 6–8]. In this paper we consider a generalization of the SPP in which the edge lengths do not have fixed value but are convex functions of continuous variables representing the position of the vertices (see Figure 1). More specifically, we have a graph in which each vertex is paired with a convex set. The spatial position of a vertex is a continuous decision variable, constrained to lie in the associated convex set. The length of an edge is a generic convex function (e.g. the Euclidean distance) of the position of the vertices it connects. When looking for a path of minimum length, we then have the extra degree of freedom of optimizing the position of the vertices visited by the path. According to the literature, this problem might be classified as an SPP *with neighborhoods* (see also Section 1.1.1 below); we use the term *graph of convex sets* to highlight the crucial role that convexity will play in the developments of this paper.

SPPs in graphs of convex sets emerge naturally in many areas. The scenario depicted in Figure 1, for example, might represent a drone flying from a source region X_s to a target region X_t . The goal is to minimize the total length of the flight. Autonomy constraints might require the drone to stop multiple times along the way, and recharging breaks can only be taken in suitable areas (convex regions). Logistic constraints, such as preventing undesired transitions between certain pairs of regions, can be embedded in the graph underlying the SPP. A main application of the proposed framework is

*Corresponding author is Tobia Marcucci (email: tobiam@mit.edu).

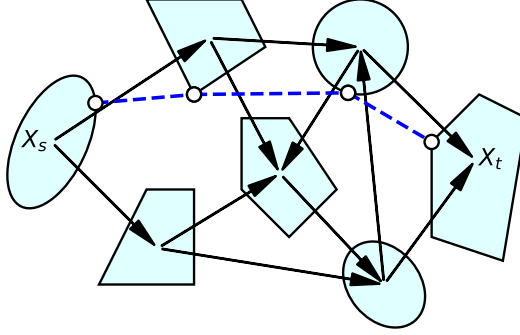


Figure 1: Example of an SPP in a graph of convex sets. The dashed blue line is the shortest path from the source set X_s to the target set X_t . The position of each vertex along the path (white circles) is allowed to move within the corresponding convex set. Transitions are allowed only between sets connected by an edge (black arrows). The cost of traveling along an edge is a convex function of the position of the vertices that this edge connects.

mixed-integer motion planning for robots [19, 20, 63] and autonomous vehicles [79, 29, 78, 92, 50] (see also the recent review [46]). In this case, the convex sets in our graph represent regions of space that do not intersect with obstacles. More abstractly, even optimal-control problems for hybrid dynamical systems [4] can be transcribed as SPPs (see Section 9).

The SPP under analysis is easily seen to be NP-hard, therefore we do not expect to find an exact polynomial-time algorithm for its solution. In this paper we relax the requirement of polynomial-time solvability and we formulate the problem as a strong Mixed-Integer Convex Program (MICP) that can be effectively solved to global optimality via branch and bound. To formulate this MICP, we develop a novel convex relaxation for a class of bilinear constraints that emerge naturally in our problem. This relaxation has its roots in the Reformulation-Linearization Technique (RLT) for bilinear programming [86] and it uses perspective functions, a tool from convex analysis that in the recent years has seen a multitude of applications in mixed-integer programming [13, 34, 39, 40, 68, 61].

Numerical results show that our MICP has a very tight convex relaxation which enables a quick identification of globally-optimal shortest paths, even when working in high-dimensional spaces and with large graphs. At present, computation times are the main limitation to a widespread application of mixed-integer motion-planning and control algorithms [71, 89, 61, 43, 62, 56]. Our MICP is substantially different from state-of-the-art formulations of these problems as we do not use binary variables to encode the region of space in which the system is at each time step, but, instead, we use binaries to select the transitions between the regions. This different parameterization yields slightly larger but much stronger MICPs that, in our computational experience, are orders of magnitude faster to solve.

Finally, moving beyond the SPP, we highlight that variants of other classical graph problems where vertices are allowed to move continuously within corresponding sets (also known as graph problems with neighborhoods) have been analyzed in the literature before (see the related works in Section 1.1.1). Exact methods for their solution generally rely on Mixed-Integer NonConvex Programming (MINCP) and are limited to very low-dimensional spaces. As we discuss in Section 10 although the techniques we propose in this paper are particularly well suited to the structure of the SPP, they apply without modification to a wide variety of graph problems with neighborhoods, yielding exact MICP formulations that have the potential to strongly outperform existing methods.

1.1 Related Works

In this subsection we overview a few variants of classical graph problems that are closely related to our problem formulation.

1.1.1 Graph Problems with Neighborhoods

As mentioned, graph problems where the vertices are allowed to move within corresponding sets are often called problems with neighborhoods¹. The SPP with neighborhoods has been analyzed in [25] under stringent assumptions that ensure polynomial-time solvability: the sets are disjoint rectilinear polygons in the plane, and the edge lengths penalize the L_1 distance between the vertices. The applications we target with this paper, however, do not verify the majority of these hypotheses. A well-studied special case of the SPP with neighborhoods is the touring-polygon problem which, in its unconstrained version, requires finding the shortest path between two points that visits a set of polygons in a given order [26]. In case of convex polygons, this problem is easily solved using convex optimization, whereas, for nonconvex polygons, it is NP-hard [26, Theorem 6]. Similar in spirit are also some classical problems in computational geometry: the safari [73], the zoo-keeper [94], and the watchman-route [14] problems.

The Traveling-Salesman Problem (TSP) with neighborhoods has been originally introduced in [2]; this paper proposed a variety of approximation algorithms for simple classes of neighborhoods. In [36], the TSP with neighborhoods has been formulated as a MINCP that is solved using spatial branch and bound. However, the expensiveness of this algorithm limits the scope of this formulation to problems in at most three dimensions. The Minimum Spanning Tree Problem (MSTP) is another problem that has been thoroughly analyzed in its version with neighborhoods [96], and also for this problem existing exact formulations are based on MINCP and do not scale beyond three dimensions [6]. Using the techniques presented in this paper, under standard convexity assumptions, both the TSP and the MSTP with neighborhoods can be formulated exactly as MICPs, a much easier class of problems than MINCP. This is discussed in Section [10].

A main application of graph problems with neighborhoods is robot coverage, a consolidated subfield of robot motion planning (see [15, 35, 7] and the references therein). Exact formulations of these problems are typically based on MINCP and can only tackle very simple coverage tasks [10]. In practice, these problems are solved approximately [10]. The techniques we propose here have the potential to greatly extend the reach of exact methods also in this area.

1.1.2 Graph Problems with Clusters of Vertices

The second family of problems we mention falls under the name of generalized Steiner problems [27] or generalized network-design problems [30, 75]. These can be thought as the discrete counterpart of the graph problems with neighborhoods: the vertex set is partitioned into clusters and the problem constraints are expressed in terms of these clusters, rather than the original vertices. For example, the TSP with clusters [72, 32, 33] asks to find a shortest tour that visits each one of the clusters of vertices at least or exactly once. Analogous generalizations have been studied for many other problems, e.g.: the MSTP [70, 28, 31, 76], the vehicle routing problem [38, 77], and graph coloring [54, 22, 21]. A clustered version of the SPP has been presented in [55]: each vertex in the graph is assigned a nonnegative weight, and the total vertex weight incurred by the shortest path within each cluster must not exceed a given value. In the same paper, a pseudo-polynomial algorithm based on Dijkstra’s algorithm [24] is proposed for the solution of this problem.

¹The word “neighborhood” here should not be interpreted in the topological sense, the vertex positions are not assumed to be in the interior of the corresponding sets.

The problem we analyze in this paper can be approximated as an SPP with clusters in a natural way. In low-dimensional spaces, this approximation can be very efficient and sufficiently accurate for practical applications. However, in high dimensions, this strategy is inoperable since approximating a portion of space with a cluster of points requires, in general, an excessive (exponential) number of points.

1.1.3 Euclidean Shortest Paths

Another variant of the SPP closely related to the one analyzed in this paper is the Euclidean SPP (see the recent book [53]). This problem requires finding a continuous path that connects two points and does not collide with a collection of polygonal obstacles. In two dimensions, the shortest path is a polygonal line whose corners are vertices of the obstacles. By constructing a visibility graph, the problem is then reduced to a discrete graph search and it is solvable in polynomial time [58, 52]. In three dimensions or more this strategy breaks; in fact, the problem becomes NP-hard [12, Theorem 2.3.2]. An approximation algorithm for the three-dimensional case has been proposed [74]. Practical algorithms for the multidimensional case based on a grid-discretization of the space have been presented in [91, 49]. More recently, exact-geometry algorithms for problems of this nature have been discussed in [23], and a moment-based technique for computing Euclidean shortest paths in case of semialgebraic obstacles has been proposed in [48].

An evident difference between the Euclidean SPP and our SPP is that the first requires the identification of a continuous path. In contrast, we only require a finite set of points to lie in appropriate regions of space, without any conditions on the path connecting them. Another fundamental difference concerns the notion of length we employ. While Euclidean-SPP algorithms strongly exploit the metric structure of the underlying space, the notion of length we use here is much weaker: a convex function with extended real values. As an example, this allows us to define the distance between two points as the energy consumed by a dynamical system to move between them, with the length being infinite in case the motion is infeasible (see Section 9).

1.2 Article Organization

This paper is organized as follows. We start in Section 2 by giving a formal statement of the problem of finding a shortest path in a graph of convex sets. Section 3 provides a detailed complexity analysis of this problem. In Section 4 we briefly recall how the classical SPP can be modeled as a Linear Program (LP): this LP will be the starting point for the derivation of our MICP formulation of the proposed SPP in Section 5. In Section 6 we provide explicit expressions for the components of the MICP in case of common types of edge-length functions and convex sets. In Section 7 we describe in more abstract terms the procedure employed to design the MICP, and we conduct a deeper technical analysis of the properties of our formulation. Except for Section 10, the rest of the paper does not depend on Section 7. The dual problem of the MICP is presented in Section 8 and it is used to prove simple bounds on the tightness of our formulation. Section 9 shows the applicability of the proposed framework to optimal control of hybrid dynamical systems. In Section 10 we discuss how other classical graph problems can be generalized from conventional graphs to graphs of convex sets. Numerical results are presented in Section 11, while conclusions and future works are in Section 12. In Appendix A we describe two other natural MICP formulations of the SPP and we compare their performance with the proposed one. Finally, proofs whose content is not relevant for the main body of the paper are deferred to Appendix B.

2 Problem Statement

We start with a formal statement of the SPP we study in this paper. Let $G := (V, E)$ be a directed graph with vertex set V and edge set $E \subseteq V^2$. For each vertex $v \in V$, we have a nonempty compact convex set $X_v \subset \mathbb{R}^d$ and a point x_v contained in it.² The length of an edge $e = (u, v)$ is determined by the location of the points x_u and x_v via the expression $\ell_e(x_u, x_v)$, where ℓ_e is convex function taking values in $\mathbb{R}_{\geq 0} \cup \{\infty\}$. We assume the functions ℓ_e to be proper (ℓ_e attains finite value in at least one point) and closed (the sublevel sets of ℓ_e are closed). Given a source vertex $s \in V$ and a target $t \in V - \{s\}$, an s - t path π is a sequence of distinct vertices $(v_k)_{k=0}^K$ such that $v_0 = s$, $v_K = t$, and $(v_k, v_{k+1}) \in E$ for all $k = 0, \dots, K-1$. We collect in the vector $x_\pi := (x_{v_k})_{k=0}^K$ the locations of the vertices visited by the path π , and we denote with $E_\pi := \{(v_k, v_{k+1})\}_{k=0}^{K-1}$ the set of traversed edges. The length of π is then defined as the sum of the length of the traversed edges:

$$\ell_\pi(x_\pi) := \sum_{e=(u,v) \in E_\pi} \ell_e(x_u, x_v).$$

Within the set Π of all s - t paths, we seek one of minimum length and, in doing this, we are allowed to optimize the vertex locations x_v . Defining $X_\pi := X_{v_0} \times \dots \times X_{v_K}$, our SPP can be compactly stated as

$$\min_{\pi \in \Pi} \min_{x_\pi \in X_\pi} \ell_\pi(x_\pi). \quad (1)$$

Remark 1. Even though we refer to the functions ℓ_e as the “edge lengths,” we underline that these need not to be valid metrics, and axioms such as symmetry or the triangle inequality are not required to hold.

Remark 2. To allow the solution of the SPP (1) to visit the same region X_v twice we can proceed as follows. We introduce an auxiliary vertex v' and we pair it with the convex set $X_{v'} := X_v$. Then we duplicate each edge (u, v) incoming v by adding (u, v') to the edge set E , and we do the same for the edges outgoing v . To allow a self-transition we can also add the edge (v, v') . Of course, this process generalizes to the case in which we want to allow a finite number of visits to the same set.

Problem (1) generalizes the classical single-source single-target SPP with nonnegative edge costs c_e : this is recovered when the edge-length functions take constant value $\ell_e(x_u, x_v) := c_e$, or when the sets X_v are singletons $\{\theta_v \in \mathbb{R}^d\}$ such that $\ell_e(\theta_u, \theta_v) = c_e$. However, it is the wide choice of edge lengths that motivates the study of the SPP (1). For example, an edge length that is commonly encountered in practice is the Euclidean distance

$$\ell_e(x_u, x_v) := \|x_v - x_u\|_2. \quad (2)$$

With this choice, the optimal location of the points x_π will define a polygonal line connecting x_s to x_t via a path as straight as possible, perfectly straight if $(s, t) \in E$. Conversely, by letting the edge length be the Euclidean distance squared

$$\ell_e(x_u, x_v) := \|x_v - x_u\|_2^2, \quad (3)$$

²For simplicity, we assume all the sets X_v to live in the same space, $\mathbb{R}^d$, but the results presented in this paper can be easily extended to the case in which each set X_v has different dimension.

straight paths may be suboptimal if they require long steps $x_v - x_u$. By allowing ℓ_e to take infinite values, we can enforce hard constraints that couple the position of the points x_u and x_v (see Section 6.1.4). This can be used to model the scenario in which the points x_v describe the time evolution a dynamical system, allowing one to formulate optimal-control problems as SPPs (see Section 9).

3 Complexity Analysis

The SPP with nonnegative edge lengths can be solved in polynomial-time using, e.g., Dijkstra's algorithm [24]. Here we show that the same cannot be expected for the SPP presented in this paper. We give two proofs of NP-hardness of problem (1). The first is very simple, and we will refer to it multiple times when analyzing the strength of our MICP. However, this proof leverages the presence of cycles in the graph G and a particular choice of the edge lengths ℓ_e . The second proof follows the complexity analysis of the Euclidean SPP presented in [12]: it covers a broader spectrum of problem instances but it is much more convoluted.

Recall that an s - t path $\pi := (v_k)_{k=0}^K$ is called Hamiltonian if it visits each vertex in the graph (i.e. if $K = |V| - 1$), and a graph is said to be Hamiltonian if it contains such a path. The Hamiltonian-Path Problem (HPP) asks if a graph G is Hamiltonian. As an example, the graph in Figure 1 is not Hamiltonian.

Theorem 1. *Problem (1) is NP-hard.*

Proof. We show that the HPP is reducible to the SPP (1) in polynomial time. The thesis then follows since the HPP is NP-complete [47].

We construct an instance of problem (1) which shares the same graph G as the given HPP. We let the source $X_s := \{0\}$ and target $X_t := \{1\}$ sets be single points on the real line, while we set $X_v := [0, 1]$ for all $v \in V - \{s, t\}$. The length of each edge is the Euclidean distance squared [3]. For a fixed path π , the optimal arrangement of the points x_π is unique and given by $x_{v_k} = k/K$ for $k = 0, \dots, K$. This yields a path length of $K(1/K)^2 = 1/K$. An optimal path is then one for which K is maximized, and it is Hamiltonian if and only if G is Hamiltonian. Synthesizing this instance, as well as verifying if $K = |V| - 1$, takes polynomial time. $\square$

Remark 3. Some graph problems with neighborhoods, e.g. the SPP analyzed in [25], are easier when the neighborhoods are disjoint. This is not the case for problem (1), which is still NP-hard under the assumption of disjoint sets X_v . In fact, in the proof of Theorem 1 the one-dimensional sets X_v , with $v \notin \{s, t\}$, can be embedded in a two-dimensional space and separated one from the others by a small gap along the second dimension, i.e.: $X_v := [0, 1] \times \{\varepsilon_v\}$ with $\varepsilon_v > 0$ very small and $\varepsilon_u \neq \varepsilon_v$ if $u \neq v$. Such a modification ensures that $X_u \cap X_v = \emptyset$, but does not affect the optimal path, which is still Hamiltonian if and only if G is Hamiltonian.

The reduction in Theorem 1 leaves two main questions open:

- For an acyclic graph G , the HPP is solvable in linear time through a topological sort [1 Section 4.4]. One might then wonder if, for an acyclic graph, problem (1) is also solvable in polynomial time. This question is of fundamental relevance for Section 9 where we show that a large family of optimal-control problems can be recast into an SPP (1) over an acyclic graph.
- Even though any strictly convex function of $x_v - x_u$ can take the place of the edge length [3] in the proof of Theorem 1 edge lengths such as (2) would not have the same effect of forcing the optimal path π to visit as many vertices as possible. One might then ask if the hardness of our problem is a consequence of the particular choice of edge lengths made in Theorem 1.

The following theorem shows that both these questions have negative answer.

Theorem 2. Assume the graph G to be acyclic, and define the edge lengths ℓ_e as in (2) for all $e \in E$. Problem (1) is NP-hard.

Sketch of proof. The proof of this theorem is quite involved, and requires retracing step by step the complexity analysis of the Euclidean SPP given in [12]. The main adjustments needed to adapt the analysis in [12] to our problem formulation are outlined in Appendix B.1 $\square$

To sum up, the SPP (1) is easily solvable when the sets X_v collapse to points or the edge lengths ℓ_e are constants. In case of a graph with cycles, large sets X_v and edge lengths ℓ_e that penalize the distance $x_v - x_u$ nonhomogeneously can make the search for a shortest path very hard. Furthermore, even in absence of these issues, hard instances of problem (1) can still be generated via a careful design of the sets X_v as the one described in [12].

4 Linear-Programming Formulation of the Classical Shortest-Path Problem

In this section we revise how the classical SPP can be modeled as an LP, and we briefly recall some important properties of this program. This will set the stage for the design of our MICP formulation of the SPP (1) in Section 5.

We use the binary variables $\{\varphi_e\}_{e \in E}$ to parameterize an s - t path in the graph G : the role of φ_e is to take unit value if and only if edge e is traversed by the path. To determine the constraints that these binaries must verify, it is convenient to interpret the SPP as the problem of shipping one unit of flow as cheaply as possible from vertex s to vertex t . From this perspective, the variable φ_e represents the units of flows carried by the edge e . Let $E_v^{\text{in}} := \{e \in E : e = (u, v) \text{ for some } u\}$, $E_v^{\text{out}} := \{e \in E : e = (v, u) \text{ for some } u\}$, and $E_v := E_v^{\text{in}} \cup E_v^{\text{out}}$ denote the sets of edges entering, leaving, and incident with vertex $v \in V$, respectively. Furthermore, for two vertices u and v , define $\delta_{uv} := 1$ if $u = v$ and $\delta_{uv} := 0$ if $u \neq v$. The flows $\varphi_v := (\varphi_e)_{e \in E_v}$ incident with vertex v are recognized to lie in the *local flow polytope*:

$$\Phi_v := \left\{ \varphi_v \stackrel{(4a)}{\geq} 0 : \sum_{e \in E_v^{\text{in}}} \varphi_e + \delta_{sv} \stackrel{(4b)}{=} \sum_{e \in E_v^{\text{out}}} \varphi_e + \delta_{tv} \stackrel{(4c)}{\leq} 1 \right\}. \quad (4)$$

Condition (4a) simply requires the flows to be nonnegative. Condition (4b) enforces the *conservation of flow*: one unit of flow is injected from the source and ejected from the target, while the incoming and outgoing flows coincide for all the other vertices. This guarantees that the edges for which $\varphi_e = 1$ actually connect the source to the target. Condition (4c) is a *degree constraint* which enforces a limit of one unit on the total flow traversing vertex v . This ensures that a path does not visit the same vertex multiple times.

Under the assumption that the edge costs c_e are nonnegative, and finite, the classical SPP can compactly formulated as the LP

$$\text{minimize} \quad \sum_{e \in E} c_e \varphi_e \quad (5a)$$

$$\text{subject to} \quad \varphi_v \in \Phi_v, \quad \forall v \in V. \quad (5b)$$

Remark 4. Note that we do not explicitly require the flow variables φ_e to be binary. This because all the basic feasible solutions of this LP can be shown to have binary value, and the addition of the constraints $\varphi_e \in \{0, 1\}$ would not affect the optimal value.

Remark 5. At the current stage, the degree constraint (4c) is redundant for problem (5). In fact, the assumption $c_e \geq 0$ makes the cost of any cycle nonnegative, and (4c) cannot be active at optimality. However, this constraint will not be redundant for the MICP we design in the next section; for this reason we include it in our problem formulation from the beginning.

4.1 Extreme Points of the Local Flow Polytopes

The extreme points $\text{ext}(\Phi_v)$ of the local flow polytopes Φ_v will play an important role in the design and the analysis of our MICP. These points are listed and described in words below; the proof that their convex hull is actually Φ_v can be found in Appendix B.2. In the following we let Δ_n denote the set of the n standard basis vectors in $\mathbb{R}^n$.

- For the source vertex s , we have $\text{ext}(\Phi_s) = \{0_{|E_s^{\text{in}}|}\} \times \Delta_{|E_s^{\text{out}}|}$. In words, all the flows incoming to s are zero, while the unit of flow that is injected in s is channeled in any of its $|E_s^{\text{out}}|$ outgoing edges.
- Symmetrically, the extreme points of the target polytope are $\text{ext}(\Phi_t) = \Delta_{|E_t^{\text{in}}|} \times \{0_{|E_t^{\text{out}}|}\}$: a unit of flow enters t from any edge, while all the outgoing flows are zero.
- For the remaining vertices $v \in V - \{s, t\}$, we have $\text{ext}(\Phi_v) = \{0_{|E_v|}\} \cup (\Delta_{|E_v^{\text{in}}|} \times \Delta_{|E_v^{\text{out}}|})$. Here we have two options: either the flow incident with v is zero, or a unit of flow is deflected from any of the incoming edges to any of the outgoing edges.

Remark 6. It is easily verified that the extreme points listed above are the only flows with binary entries contained in the polytopes Φ_v . Mathematically, $\text{ext}(\Phi_v) = \Phi_v \cap \{0, 1\}^{|E_v|}$ for all $v \in V$.

5 Mixed-Integer Convex Formulation

We formulate the SPP (1) as an MICP that can be efficiently solved to global optimality via branch and bound. We do this in three steps:

1. In Section 5.1 we extend the LP formulation (5) of the classical SPP to our generalized setting: this step involves perspective functions and yields an optimization with bilinear equality constraints.
2. In Section 5.2 we use the perspective operation a second time to convexify the constraints of this bilinear program and design a first MICP formulation of the SPP (1).
3. In Section 5.3 we show that part of the decision variables and constraints of this MICP are redundant. This results in a reduced-size optimization that will be our definitive MICP formulation of the SPP (1).

All the necessary background on perspective functions is introduced. Explicit expressions of the various components of the MICP are given in Section 6 for common edge lengths ℓ_e and convex sets X_v . An in-depth analysis of the properties of the proposed MICP can be found in Section 7.

5.1 Bilinear Formulation

We build on the LP (5) to formulate the SPP (1) as a mathematical program. A natural attempt in this direction is to proceed as done for many other graph problems with neighborhoods [36, 6, 10]: include the vertex positions x_v among the decision variables of (5), enforce the convex constraints $x_v \in X_v$ for all $v \in V$, and substitute the objective (5a) with the nonconvex function

$$\sum_{e=(u,v) \in E} \ell_e(x_u, x_v) \varphi_e. \quad (6)$$

Unfortunately, this approach has two flaws. The first, and most important, is that the nonconvexity of (6) makes this program very hard to solve. The second is more of a technicality: since in our problem formulation the functions ℓ_e are allowed to take infinite value, the product in (6) is undefined if $\ell_e(x_u, x_v) = \infty$ and $\varphi_e = 0$, while for $\varphi_e = 0$ we would always want the e th cost addend to be zero. The first issue will be the subject of the next subsection: there we will see that, if the flow variables φ_e are actually required to take binary values, our mathematical program can be convexified exactly. In this subsection we show how the desired behavior of “turning on and off” the edge lengths ℓ_e with the flows φ_e can be correctly achieved using perspective functions [44, Section IV.2.2].

The next definition might appear cumbersome for numerical calculations but, as shown in Section 6, in most common cases it yields simple expressions readily amenable to standard optimization solvers.

Definition 1. Let $f : \mathbb{R}^n \rightarrow \mathbb{R} \cup \{\infty\}$ be a closed convex function, and let $\bar{x} \in \mathbb{R}^n$ be any point such that $f(\bar{x})$ is finite. We define the *perspective*³ of the function f as

$$\tilde{f}(x, \lambda) := \begin{cases} \lambda f(x/\lambda) & \text{if } \lambda > 0 \\ \lim_{\tau \downarrow 0} \tau f(\bar{x} + x/\tau) & \text{if } \lambda = 0, \\ \infty & \text{if } \lambda < 0 \end{cases}$$

where the value of the limit operation can be shown to be independent of the point $\bar{x}$.

A fundamental property of the perspective operation is that it preserves convexity [44, Propositions IV.2.2.1 and IV.2.2.2]: the function $\tilde{f}(x, \lambda)$ is jointly convex in x and λ . Before moving on, let us work out two simple examples to get familiar with Definition 1.

Example 1. Consider the norm $f(x) := \|x\|$. For $\lambda > 0$ its perspective is $\tilde{f}(x, \lambda) := \lambda \|x/\lambda\| = \|x\|$. For $\lambda = 0$, we set $\bar{x} = 0$ and we get $\tilde{f}(x, 0) := \lim_{\tau \downarrow 0} \tau \|x/\tau\| = \lim_{\tau \downarrow 0} \|x\| = \|x\|$. Overall, we then have

$$\tilde{f}(x, \lambda) = \begin{cases} \|x\| & \text{if } \lambda \geq 0 \\ \infty & \text{if } \lambda < 0 \end{cases}.$$

The same logic shows that any positively homogeneous function f (i.e., any function such that $f(cx) = cf(x)$ for all x and $c > 0$) is such that $\tilde{f}(x, \lambda) = f(x)$ if $\lambda \geq 0$ and $\tilde{f}(x, \lambda) = \infty$ if $\lambda < 0$.

Example 2. Let $f(x) := \|x\|_2^2$. For $\lambda > 0$ the perspective function is $\tilde{f}(x, \lambda) := \lambda \|x/\lambda\|_2^2 = \|x\|_2^2/\lambda$. For $\lambda = 0$, letting $\bar{x} = 0$, we get $\tilde{f}(x, 0) := \lim_{\tau \downarrow 0} \tau \|x/\tau\|_2^2 = \lim_{\tau \downarrow 0} \|x\|_2^2/\tau$. This limit evaluates to zero if $x = 0$ and to infinity if $x \neq 0$. Therefore,

$$\tilde{f}(x, \lambda) := \begin{cases} \|x\|_2^2/\lambda & \text{if } \lambda > 0 \\ 0 & \text{if } \lambda = 0 \text{ and } x = 0. \\ \infty & \text{otherwise} \end{cases}$$

³More precisely, the one defined is the closure of the perspective function of f ; the perspective function is typically defined to be infinite for $\lambda = 0$ [44, Section IV.2.2]. Since in this paper we only work with the former, there is no risk of misunderstanding.

With the perspective operation at our disposal, we go back to our SPP. We define two auxiliary variables per edge

$$y_e := \varphi_e x_u, \quad z_e := \varphi_e x_v, \quad \forall e = (u, v) \in E, \quad (7)$$

and we replace the objective (6) with the convex function⁴

$$\sum_{e \in E} \tilde{\ell}_e(y_e, z_e, \varphi_e). \quad (8)$$

When a flow variable φ_e is positive, the expression in (8) is obtained from (6) simply by dividing and multiplying the arguments of ℓ_e by φ_e :

$$\ell_e(x_u, x_v) \varphi_e = \ell_e(\varphi_e x_u / \varphi_e, \varphi_e x_v / \varphi_e) \varphi_e = \ell_e(y_e / \varphi_e, z_e / \varphi_e) \varphi_e =: \tilde{\ell}_e(y_e, z_e, \varphi_e). \quad (9)$$

When $\varphi_e = 0$ the e th addend in (8) is always well defined, and it correctly evaluates to zero, even if $\ell_e(x_u, x_v) = \infty$. In fact, $\varphi_e = 0$ implies $y_e = z_e = 0$ by (7), and Definition 1 gives

$$\tilde{\ell}_e(0, 0, 0) := \lim_{\tau \downarrow 0} \tau \ell_e(\bar{x}_u + 0/\tau, \bar{x}_v + 0/\tau) = \lim_{\tau \downarrow 0} \tau \ell_e(\bar{x}_u, \bar{x}_v) = 0,$$

where $\bar{x}_u$ and $\bar{x}_v$ are any two points such that $\ell_e(\bar{x}_u, \bar{x}_v)$ is finite.

Overall, we then have the following formulation of the SPP (1):

$$\text{minimize} \quad \sum_{e \in E} \tilde{\ell}_e(y_e, z_e, \varphi_e) \quad (10a)$$

$$\text{subject to} \quad \varphi_v \in \Phi_v, \quad x_v \in X_v, \quad \forall v \in V, \quad (10b)$$

$$y_e = \varphi_e x_u, \quad z_e = \varphi_e x_v, \quad \forall e = (u, v) \in E. \quad (10c)$$

The decision variables are the flows φ_e , the vertex positions x_v , and the auxiliary variables y_e and z_e . The role of the latter is to match the vertices x_u and x_v when the edge $e = (u, v)$ is traversed by a unit of flow, and collapse to zero when $\varphi_e = 0$. This behavior is driven by the bilinear equality constraints (10c), which are the only nonconvexity in our formulation and whose convexification is the focus of the next subsection. Before that, let us formally verify that the integrality property of the LP (5), discussed in Remark 4, is actually inherited by the bilinear program (10).

Proposition 1. *Let ℓ^* be a local minimum of problem (10). There exists a feasible assignment for the variables of this problem with cost equal to ℓ^* and such that $\varphi_e \in \{0, 1\}$ for all $e \in E$.*

Proof. Let $\{x_v^*\}_{v \in V}$ and $\{\varphi_e^*\}_{e \in E}$ be a local minimizer of problem (10) with cost ℓ^* . We add to (10) the constraints $x_v = x_v^*$ for all $v \in V$ and $\varphi_e = 0$ for all $e \in E$ such that $\varphi_e^* = 0$. After a few manipulations, this program is simplified to an LP of the form (5) with edge set $E' := \{e \in E : \varphi_e^* > 0\}$ and edge costs $c_e := \ell_e(x_u^*, x_v^*)$ for all $e \in E'$. Note that these costs are finite since $\ell_e(x_u^*, x_v^*) = \infty$ and $\varphi_e^* > 0$ would imply $\ell^* = \infty$ by (9) and (10a). The optimal value of this LP must equal ℓ^* , otherwise our solution of (10) would not be optimal, not even locally. Furthermore, because of the integrality property from Remark 4, we can assume the optimal flows of this LP to be binary. Paired with the variables x_v^* , these binary flows yield a feasible solution of (10) with cost ℓ^* . $\square$

In simpler words, Proposition 1 tells us that if we manage to solve the bilinear program (10), even only to local optimality, a solution with integral flows φ_e and same cost as the one we have found can always be recovered very cheaply.

⁴Here we are slightly abusing notation: since we defined the perspective operation for functions with a single argument, to be precise, we should write $\ell_e((y_e, z_e), \varphi_e)$.

5.2 Mixed-Integer Convex Reformulation of the Bilinear Program

Problem (10) is our first formulation of the SPP (1) that can be solved with a computer. Unfortunately, the bilinear equalities (10c) make this optimization prohibitive: even for the simplest edge lengths ℓ_e and convex sets X_v , the most effective solvers we have to tackle a problem of this kind are local methods with no convergence guarantees (e.g. [93]). Our next step is then to reformulate this program as an MICP that can be reliably solved to global optimality using branch and bound.

We seek a *mixed-integer convex formulation* of the feasible set⁵ of problem (10). With this we mean a family of constraints on the decision variables of (10) that meets two requirements:

1. it delimits a convex set when the flow variables φ_e are allowed to take fractional values,
2. it correctly enforces the constraints of problem (10) when the flows φ_e are binary.

In doing this we are required to tradeoff the *size* of the formulation (number of constraints and, possibly, auxiliary variables) with the *strength* of the formulation (tightness with which the convex set we design envelops the feasible set of the bilinear program). The approach we describe below represents, in our experience, the best compromise between these two conflicting needs. For completeness, in Appendix A we describe two other natural candidate formulations and we provide numerical evidence of the performance gap between them and the proposed method.

We start by introducing a second perspective operation, this time applied to sets.

Definition 2. Let $S \subset \mathbb{R}^n$ be a compact⁶ convex set. We define the *perspective* of S as the set

$$\tilde{S} := \{(x, \lambda) \in \mathbb{R}^{n+1} : \lambda \geq 0, x \in \lambda S\}. \quad (11)$$

Just as its counterpart from Definition 1 the set perspective preserves convexity: the set $\tilde{S}$ is easily verified to be a pointed closed convex cone.

Example 3. Consider the unit ball $S := \{x : \|x\| \leq 1\}$. For $\lambda > 0$, we have $\lambda S = \{\lambda x : \|x\| \leq 1\} = \{x : \|x/\lambda\| \leq 1\} = \{x : \|x\| \leq \lambda\}$. This equality holds also for $\lambda = 0$, in fact, $0S = \{0\} = \{x : \|x\| \leq 0\}$. The perspective of the unit ball is then the so-called *norm cone* [9, Section 2.2.3]: $\tilde{S} := \{(x, \lambda) : \|x\| \leq \lambda\}$.

Remark 7. The set $\tilde{S}$ is commonly encountered in convex analysis (see e.g. [81, Section 8] or [44, Section V.1.2]), and sometimes it is referred to as the *cone over S* . Here we call it *perspective* of S to emphasize the connection between the operations in Definitions 1 and 2. Let us denote with ι_S the *indicator function* of S : $\iota_S(x) := 0$ if $x \in S$ and $\iota_S(x) := \infty$ if $x \notin S$. It is easily verified that the functions $\iota_{\tilde{S}}$ and $\tilde{\iota}_S$ coincide, i.e., the set perspective morphs the set S the same way as the function perspective morphs its indicator function.

The set-perspective operation is our main tool to design a convex envelope around the feasible set of the bilinear program (10). We do this in an algorithmic fashion. Let $a^\top \varphi_v + b$ be any linear function of the flows incident with vertex v . Multiplying this function by the vertex position x_v , and expanding the product, we get the identity $(a^\top \varphi_v + b)x_v = (x_v \varphi_v^\top) a + b x_v$. We then define $M_v \in \mathbb{R}^{d \times |E_v|}$ as the matrix whose columns are the auxiliary variables z_e for $e \in E_v^{\text{in}}$ and y_e for $e \in E_v^{\text{out}}$, and we note that,

⁵For an optimization problem, we call *feasible set* the set of decision variables that verify all the constraints.

⁶The properties of the set $\tilde{S}$ that we discuss in this paper hold also for an unbounded set S , provided that in (11) we let $x \in \lambda S + S_\infty$, with S_∞ denoting the recession cone of S [81, Section 8]. Note that this definition generalizes the one we give, since for a bounded set we have $S_\infty = \{0\}$. To unburden the presentation of technical details, and since all the sets we work with in this paper are bounded, we define the set-perspective operation only for bounded sets.

in terms of this matrix, the bilinear constraints (10c) read $M_v = x_v \varphi_v^\top$. We conclude that the following nonlinear equality is valid⁷ for the feasible set of the bilinear program (10):

$$(a^\top \varphi_v + b)x_v = M_v a + b x_v. \quad (12)$$

This suggests the following lemma.

Lemma 1. *Let $a^\top \varphi_v + b$ be a linear function of the flows incident with vertex $v \in V$.*

- (a) *If the equality $a^\top \varphi_v + b = 0$ is valid for the feasible set of program (10), so is the linear equality constraint*

$$\begin{bmatrix} M_v a + b x_v \\ a^\top \varphi_v + b \end{bmatrix} = 0. \quad (13)$$

- (b) *If the inequality $a^\top \varphi_v + b \geq 0$ is valid for the feasible set of program (10), so is the convex constraint*

$$\begin{bmatrix} M_v a + b x_v \\ a^\top \varphi_v + b \end{bmatrix} \in \tilde{X}_v. \quad (14)$$

Proof. Point (a) follows immediately from (12) and the assumption $a^\top \varphi_v + b = 0$. For point (b), note that constraint (14) enforces two conditions: $a^\top \varphi_v + b \geq 0$ and $M_v a + b x_v \in (a^\top \varphi_v + b)X_v$. The first is assumed, the second follows from (12) and the constraint $x_v \in X_v$ in (10b). Finally, the convexity of (14) is due to the convexity of $\tilde{X}_v$ and to the linearity of the vector on the left-hand side in the decision variables of (10). $\square$

Lemma 1 translates any valid linear constraint on the flows φ_v into a convex constraint that envelops the feasible set of problem (10). Our mixed-integer convex formulation is obtained simply by applying this lemma to each one of the constraints defining the local flow polytope Φ_v in (4).

- We start by applying Lemma 1(b) to the nonnegativity constraint (4a). In this case, the vector a is any of the standard basis vectors of dimension $|E_v|$, while the scalar b is zero. Thus the product $M_v a$ in (14) selects one of the auxiliary variable z_e for $e \in E_v^{\text{in}}$ or y_e for $e \in E_v^{\text{out}}$. The resulting convex constraints are

$$(z_e, \varphi_e) \in \tilde{X}_v, \quad \forall e \in E_v^{\text{in}}, \quad (15a)$$

$$(y_e, \varphi_e) \in \tilde{X}_v, \quad \forall e \in E_v^{\text{out}}. \quad (15b)$$

- Consider the conservation of flow (4b). To put this constraint in the form $a^\top \varphi_v + b = 0$ we define $a := (1_{|E_v^{\text{in}}|}, -1_{|E_v^{\text{out}}|})$ and $b := \delta_{sv} - \delta_{tv}$, where 1_n denotes a vector with n entries equal to one, and where the first entries in φ_v are assumed to be the flows incoming v and the last entries to be the outgoing ones. Lemma 1(a) gives us then the linear constraint

$$\begin{bmatrix} \sum_{e \in E_v^{\text{in}}} z_e - \sum_{e \in E_v^{\text{out}}} y_e + \delta_{sv} x_v - \delta_{tv} x_v \\ \sum_{e \in E_v^{\text{in}}} \varphi_e - \sum_{e \in E_v^{\text{out}}} \varphi_e + \delta_{sv} - \delta_{tv} \end{bmatrix} = 0. \quad (16)$$

⁷Recall also that a constraint that is verified by all the points in a set is said to be *valid* for that set.

- Finally, the degree constraint (4c). Applying Lemma 1(b) with $a := (0_{|E_v^{\text{in}}|}, -1_{|E_v^{\text{out}}|})$ and $b := 1 - \delta_{tv}$, we get the convex constraint

$$\begin{bmatrix} (1 - \delta_{tv})x_v - \sum_{e \in E_v^{\text{out}}} y_e \\ 1 - \delta_{tv} - \sum_{e \in E_v^{\text{out}}} \varphi_e \end{bmatrix} \in \tilde{X}_v. \quad (17)$$

Out of the two requisites listed above for a mixed-integer convex formulation, the three constraints we just derived certainly meet the first (convexity). What is less obvious is that they are also sufficient to fulfill the second, and correctly replace the constraints of problem (10) in case of a binary flow. This is shown in Theorem 3 below. Before that, let us explicitly state the MICP formulation of the SPP (1) that results from this substitution:

$$\text{minimize} \quad \sum_{e \in E} \tilde{\ell}_e(y_e, z_e, \varphi_e) \quad (18a)$$

$$\text{subject to} \quad \text{nonnegativity constraint (15),} \quad \forall v \in V, \quad (18b)$$

$$\text{conservations of flow (16),} \quad \forall v \in V, \quad (18c)$$

$$\text{degree constraint (17),} \quad \forall v \in V, \quad (18d)$$

$$\varphi_e \in \{0, 1\}, \quad \forall e \in E. \quad (18e)$$

This program shares the same decision variables as (10): φ_e, y_e, z_e , as well as the vertex positions x_v . Its convex relaxation is obtained simply by dropping constraint (18e). We will call *relaxation gap* the difference between the optimal value of an MICP and of its convex relaxation. Note that, in contrast to the bilinear formulation, we generally expect the optimal value of this MICP to decrease when the flow variables φ_e are allowed to take fractional values.

Theorem 3. *The feasible set of the bilinear program (10), subject to the additional constraints $\varphi_e \in \{0, 1\}$ for all $e \in E$, coincides with the feasible set of the MICP (18).*

We will see in Section 7.2.2 that this theorem follows from a simple geometric property of Lemma 1. Here we give a direct proof of this result; this will also help us gaining a better understanding of the logic behind our formulation.

Proof of Theorem 3. By construction, any feasible solution of the bilinear program with integer flows verifies the constraints of the MICP. Therefore we only have to show the reverse inclusion.

Assume we are given a feasible point for the MICP. We start by considering the source vertex s . Distributed between the constraints (18b)–(18d), we have the constraint $\varphi_s \in \Phi_s$. As seen in Section 4.1, this polytope, paired with the integrality condition (18e), forces the flows incoming s to be zero and exactly one of the flows outgoing s to be one. Let $o \in E_s^{\text{out}}$ be the edge such that $\varphi_o = 1$. The nonnegativity constraint (15) gives $z_e = 0$ for all $e \in E_s^{\text{in}}$, $y_e = 0$ for all $e \in E_s^{\text{out}} - \{o\}$, and $y_o \in X_s$. The conservation of flow (16) gives $x_s = y_o$. The degree constraint (17) is then redundant. This behavior agrees with the constraints of the bilinear program. The analysis for the target t is specular to the one of the source.

For $v \notin \{s, t\}$, the constraints (18b)–(18d) imply $\varphi_v \in \Phi_v$. This condition, together with (18e), forces either all the flows incident with v to be zero or one unit of flow to traverse v via two edges $i \in E_v^{\text{in}}$ and $o \in E_v^{\text{out}}$. In the zero-flow scenario, the nonnegativity constraint (15) sets the variables z_e and y_e to zero, the conservation of flow (16) holds trivially, and the degree constraint (17) ensures that $x_v \in X_v$. When v is traversed by a unit of flow, from (15) we have $z_e = 0$ for all $e \in E_v^{\text{in}} - \{i\}$, $y_e = 0$ for all $e \in E_v^{\text{out}} - \{o\}$, and $z_i, y_o \in X_v$. Then (16) and (17) give $z_i = y_o = x_v$. Both these scenarios agree with the constraints of the bilinear program. $\square$

Remark 8. Even though the constraint $x_v \in X_v$ is not explicitly enforced in the MICP (18), Theorem 3 shows that this condition is verified by any feasible solution of our program. It turns out that the same is true also for the convex relaxation of (18), and that the inclusion $x_v \in X_v$ is implied by (18b)–(18d). This will be proven at a higher level of generality in Section 7.2.1.

Remark 9. To construct our MICP we have operated on the constraints corresponding to each vertex in our graph independently. Similarly to known hierarchies in 0-1 programming [84, 57], the principle behind Lemma 1 could be extended to leverage potential coupling constraints between flows that do not share a common vertex. For example, for two vertices u and v , a linear constraint involving the flows φ_u and φ_v could be multiplied by the vertex positions x_u and x_v , and used as in Lemma 1 to tighten our convex envelope. This, however, would require the introduction of auxiliary decision variables representing the mixed products $x_v \varphi_u^\top$ and $x_u \varphi_v^\top$, and it would rapidly make the size of our formulation intractable.

5.3 Reduced Mixed-Integer Convex Formulation

Not all the components of the MICP (18) are actually necessary. In this subsection we show that, due to the particular structure of the local flow polytopes Φ_v in (4), the vertex locations x_v , together with all the constraints involving them, can be removed from our MICP without affecting its optimal value. If needed, an optimal assignment for the variables x_v can be reconstructed a posteriori from the values of the flows φ_e and the auxiliary variables y_e and z_e . This observation will significantly decrease the size and the solution time of our MICP. For a more abstract geometric analysis of this phenomenon we refer the reader to Section 7.4.

Remark 10. Let us introduce some terminology to streamline the upcoming discussion. In the rest of this paper, we will refer to the variables x_v , y_e , and z_e as the *spatial variables*, as they live in the space $\mathbb{R}^d$ underlying our SPP. A constraint of the form (13) enforces two conditions: $M_v a \varphi_v + b x_v = 0$ and $a^\top \varphi_v + b = 0$. With the term *spatial constraint* we refer to the former, since it is the one involving the spatial variables. We use the same name for the condition $M_v a + b x_v \in (a^\top \varphi_v + b) X_v$ that is enforced in (14).

When Lemma 1 is applied to a homogeneous flow constraint (i.e. a constraint with $b = 0$), the resulting convex constraint (either of the form (13) or (14)) does not directly involve the decision variable x_v . Since most of the constraints defining the polytopes Φ_v are homogeneous, the vertex positions x_v appear only in a handful of the convex constraints we derived in the previous subsection. More specifically, we notice the following:

- For a vertex $v \notin \{s, t\}$, the variable x_v appears only in the spatial degree constraint in (17), which reads $x_v - \sum_{e \in E_v^{\text{out}}} y_e \in (1 - \sum_{e \in E_v^{\text{out}}} \varphi_e) X_v$. Letting θ_v be any point in X_v , this constraint can be used to express x_v as a function of the other decision variables:

$$x_v := \sum_{e \in E_v^{\text{out}}} y_e + \left(1 - \sum_{e \in E_v^{\text{out}}} \varphi_e\right) \theta_v, \quad \forall v \in V - \{s, t\}. \quad (19)$$

The spatial degree constraint and the variable x_v can then be removed from our program.

- The spatial degree constraints in (17) are redundant also for $v \in \{s, t\}$. In fact, for the source s , the flow polytope Φ_s ensures that $\sum_{e \in E_s^{\text{out}}} \varphi_e = 1$ and the spatial conservation of flow in (16) reads $x_s = \sum_{e \in E_s^{\text{out}}} y_e$. Under these two conditions, the spatial degree constraint becomes $0 \in 0 X_s$,

which is trivially redundant. For the target t , we have $\varphi_e = 0$ and $y_e = 0$ for all $e \in E_t^{\text{out}}$. Again, the spatial degree constraint becomes $0 \in 0X_t$.

- After having removed the spatial degree constraints, the vertex positions x_s and x_t appear only in the spatial conservation of flow in (16) for $v \in \{s, t\}$. Again, we solve these two equalities out of our MICP by defining

$$x_s := \sum_{e \in E_s^{\text{out}}} y_e, \quad x_t := \sum_{e \in E_t^{\text{in}}} z_e. \quad (20)$$

These observations lead to a reduced MICP where the decision variables x_v are dropped, together with the spatial degree constraints in (17) and the spatial conservation of flow in (16) for $v \in \{s, t\}$. Expanding all the constraints, our conclusive MICP formulation of the SPP (1) reads

$$\text{minimize} \quad \sum_{e \in E} \tilde{\ell}_e(y_e, z_e, \varphi_e) \quad (21a)$$

$$\text{subject to} \quad (y_e, \varphi_e) \in \tilde{X}_u, \quad (z_e, \varphi_e) \in \tilde{X}_v, \quad \forall e = (u, v) \in E, \quad (21b)$$

$$\sum_{e \in E_v^{\text{in}}} \varphi_e + \delta_{sv} = \sum_{e \in E_v^{\text{out}}} \varphi_e + \delta_{tv} \leq 1, \quad \forall v \in V, \quad (21c)$$

$$\sum_{e \in E_v^{\text{in}}} z_e = \sum_{e \in E_v^{\text{out}}} y_e, \quad \forall v \in V - \{s, t\}, \quad (21d)$$

$$\varphi_e \in \{0, 1\}, \quad \forall e \in E. \quad (21e)$$

After having solved this program, an optimal location of the vertices is recovered using (19) and (20).

The reduced MICP (21) has $|E|$ binary variables and $2d|E|$ continuous variables. Assuming the number of constraints defining the sets $\tilde{X}_v$ to be $h(d)$, its convex relaxation (obtained by dropping (21e)) has a total of $2|V| + d(|V| - 2) + 2h(d)|E|$ constraints. As shown in the next section, $h(d)$ is typically constant or linear, and the size of our MICP scales bilinearly with the size of the graph G and the dimension d of the space in which the sets X_v live.

6 Perspective-Function Toolbox

In order to actually solve our MICP with a computer, we need implementable descriptions of the perspective functions $\tilde{\ell}_e$ in the objective (21a) and the sets $\tilde{X}_v$ in the constraint (21b). In this section we provide explicit expressions for these components of our program in case of commonly-used edge lengths ℓ_e and convex sets X_v . In addition, we show that when the former are constants or the latter are singletons, our MICP simplifies to the LP formulation (5) of the classical SPP and hence, in accordance with Remark 4, it has zero relaxation gap.

6.1 Common Choices for the Edge Lengths ℓ_e

We give explicit expressions for the perspectives $\tilde{\ell}_e$ in the objective (21a) in case of commonly-used edge lengths ℓ_e .

6.1.1 When the Edge Lengths ℓ_e are Constants

We start by showing that, when the edge lengths are finite nonnegative constants, the MICP (21) is equivalent to the LP formulation (5) of the classical SPP and hence it has zero relaxation gap.

Assume $\ell_e(x_u, x_v) := c_e \geq 0$ for all $e \in E$. Using Definition 1 the addends $\tilde{\ell}_e(y_e, z_e, \varphi_e)$ in our objective are easily verified to coincide with $c_e \varphi_e$ for $\varphi_e \geq 0$. For all $e = (u, v) \in E$, we can then define $y_e := \varphi_e \theta_u$ and $z_e := \varphi_e \theta_v$ for some $\theta_u \in X_u$ and $\theta_v \in X_v$. The spatial constraints in (21b) are verified. The spatial conservation of flow (21d) simplifies to $\sum_{e \in E_v^{\text{in}}} \varphi_e \theta_v = \sum_{e \in E_v^{\text{out}}} \varphi_e \theta_v$, and it is implied by the regular conservation of flow in (21c). The MICP is then reduced to the LP (5) with the additional integrality requirement (21e), which we know is redundant in this case (see Remark 4).

6.1.2 When the Edge Lengths ℓ_e are Positively Homogeneous

Assume the edge length ℓ_e to be positively homogeneous, i.e., $\ell_e(cx_u, cx_v) = c\ell_e(x_u, x_v)$ for all x_u, x_v , and $c > 0$. An example of such a function is $\ell_e(x_u, x_v) := \|A_e x_u + B_e x_v\|$, from which the Euclidean length (2) is recovered when the norm is the 2-norm and $B_e := -A_e := I$. As shown in Example 1, in this special case we have $\tilde{\ell}_e(y_e, z_e, \varphi_e) = \ell_e(y_e, z_e)$ for $\varphi_e \geq 0$. In case of a p -norm, using slack variables, this is implementable as a linear objective subject to linear constraints if $p \in \{1, \infty\}$ or to a Second-Order Cone Constraint (SOCC) if $p = 2$.

6.1.3 When the Edge Lengths ℓ_e are Positive-Semidefinite Quadratic Forms

Let the edge length be defined as $\ell_e(x_u, x_v) := \|A_e x_u + B_e x_v\|_2^2$. Notice that the Euclidean length squared (3) is recovered as a special case for $B_e := -A_e := I$. Proceeding as in Example 2, in this case we have

$$\tilde{\ell}_e(y_e, z_e, \varphi_e) = \begin{cases} \|A_e y_e + B_e z_e\|_2^2 / \varphi_e & \text{if } \varphi_e > 0 \\ 0 & \text{if } \varphi_e = 0 \text{ and } A_e y_e + B_e z_e = 0 \\ \infty & \text{otherwise} \end{cases} \quad (22)$$

The three cases in this equation are easily modeled via a SOCC. We introduce a nonnegative slack variable l_e that takes the place of the edge length ℓ_e in our objective. We then add the rotated SOCC

$$\varphi_e l_e \geq \|A_e y_e + B_e z_e\|_2^2. \quad (23)$$

For $\varphi_e > 0$, the slack l_e is forced by the cost to coincide with $\|A_e y_e + B_e z_e\|_2^2 / \varphi_e$, as required by (22). For $\varphi_e = 0$, the variable l_e is pushed to zero which, recalling that $\varphi_e = 0$ implies $y_e = z_e = 0$ by (21b), also agrees with (22). We conclude that the SOCC (23) models the above perspective correctly.

6.1.4 When the Edge Lengths ℓ_e Enforce Hard Constraints

Imagine we want to couple the position of the two endpoints x_u and x_v of the edge $e = (u, v)$ via a constraint of the form $(x_u, x_v) \in X_e$, with X_e nonempty, closed, and convex. Without loss of generality, we can also assume the set X_e to be bounded; if not, we can just replace it with the equivalent bounded set $X_e \cap (X_u \times X_v)$. Our goal can be achieved by defining the edge length

$$\ell_e(x_u, x_v) := \begin{cases} \ell'_e(x_u, x_v) & \text{if } (x_u, x_v) \in X_e \\ \infty & \text{otherwise} \end{cases},$$

for some suitable function ℓ'_e . After a few manipulations of Definition [1](#) the perspective of this edge length is found to be

$$\tilde{\ell}_e(y_e, z_e, \varphi_e) := \begin{cases} \tilde{\ell}'_e(y_e, z_e, \varphi_e) & \text{if } (y_e, z_e, \varphi_e) \in \tilde{X}_e \\ \infty & \text{otherwise} \end{cases}.$$

Therefore, in practice, the requirement $(x_u, x_v) \in X_e$ simply results in the hard constraint $(y_e, z_e, \varphi_e) \in \tilde{X}_e$, which can be implemented using the results from the next subsection.

6.2 Common Choices for the Convex Sets X_v

We now consider common choices for the convex sets X_v and we give descriptions of their perspectives $\tilde{X}_v$ that are readily amenable to standard solvers. The next lemma draws a parallel between function and set perspectives similar to the one discussed in Remark [7](#). It shows how a functional description of a convex set can be translated into a functional description of its perspective.

Lemma 2. *Let the functions f_i verify the conditions in Definition [1](#) for all i in some index set I . Assume the convex set $S := \{x : f_i(x) \leq 0 \text{ for all } i \in I\}$ to be bounded. We have*

$$\tilde{S} = \{(x, \lambda) : \tilde{f}_i(x, \lambda) \leq 0 \text{ for all } i \in I\}.$$

Proof. We verify that the two sets are equal when sliced for different values of λ . If $\lambda > 0$, the condition $\tilde{f}_i(x, \lambda) \leq 0$ is equivalent to $f_i(x/\lambda) \leq 0$. Enforcing this for all i is, in turn, equivalent to $x \in \lambda S$. If $\lambda = 0$, the conditions $\tilde{f}_i(x, 0) \leq 0$ read $\lim_{\tau \downarrow 0} \tau f_i(\bar{x} + x/\tau) \leq 0$. These are recognized to force x to lie in the recession cone of S . Since S is bounded, its recession cone is $\{0\}$. This agrees with $\lambda S = \{0\}$. Finally, both slices are empty when $\lambda < 0$. $\square$

Remark 11. In Lemma [2](#) the boundedness of S is necessary for the perspective function to drive x to the origin as λ goes to zero. This is the main reason why we assume the sets X_v to be bounded: otherwise, we would not have a practical way to drive the variables y_e and z_e to the origin as φ_e goes to zero, as required by constraint [\(21b\)](#).

6.2.1 When the Sets X_v are Singletons

As in the case of constant edge lengths from Section [6.1.1](#) when the sets X_v are singletons our MICP simplifies to the LP [\(5\)](#) and has zero relaxation gap.

When $X_v := \{\theta_v\}$ for all $v \in V$, the spatial constraints in [\(21b\)](#) simply become linear equalities: $y_e = \varphi_e \theta_u$ and $z_e = \varphi_e \theta_v$ for all $e = (u, v) \in E$. The conservation of flow in [\(21c\)](#) implies its spatial counterpart [\(21d\)](#). The auxiliary variables y_e and z_e can be then plugged in the objective and eliminated from the problem. The eth cost addend becomes

$$\tilde{\ell}_e(\varphi_e \theta_u, \varphi_e \theta_v, \varphi_e) = \begin{cases} \varphi_e \ell_e(\theta_u, \theta_v) & \text{if } \varphi_e > 0 \\ 0 & \text{if } \varphi_e = 0 \end{cases}.$$

Assuming each constant edge length $c_e := \ell_e(\theta_u, \theta_v)$ to be finite (otherwise we can just remove the edge e from the problem), the latter equals $c_e \varphi_e$. This reduces our MICP to the LP [\(5\)](#).

6.2.2 When the Sets X_v are Polytopes

Assume the sets X_v to be polytopes with halfspace representation $\{x : A_v x \leq b_v\}$. Let $A_{v,i}$ be the i th row of A_v and $b_{v,i}$ the i th entry of b_v . Using Definition 1, the perspective of the function $A_{v,i}x - b_{v,i}$ is easily found to be $A_{v,i}x - b_{v,i}\lambda$ for $\lambda \geq 0$. Using Lemma 2 we then have $\tilde{X}_v = \{(x, \lambda) : A_v x \leq b_v \lambda\}$. In this case, the convex constraints (21b) are enforced as linear inequalities.

6.2.3 When the Sets X_v are Affine Transformations of the Unit Ball

We conclude with a generalization of Example 3. We let $X_v := \{x : \|A_v x + b_v\| \leq 1\}$. Using Lemma 2 and Definition 1 it can be seen that $\tilde{X}_v = \{(x, \lambda) : \|A_v x + b_v \lambda\| \leq \lambda\}$. For a p -norm with $p \in \{1, \infty\}$ the convex constraints (21b) are then linear, and for $p = 2$ they are SOCCs.

7 Analysis of the Mixed-Integer Convex Formulation

In this section we analyze in greater depth the properties of the MICP we designed in Section 5. The starting point, in Section 7.1 will be to describe in abstract terms how Lemma 1 operates on the sets Φ_v and X_v to relax the feasible set of the bilinear program (10). This abstraction will shed light on multiple issues that the derivation in Section 5 left open:

- In Section 7.2.1 we will certify that, unless further assumptions on the problem structure are made, no other strengthening constraint for our MICP can be generated using Lemma 1.
- Theorem 3 showed that the convex constraints we designed for the MICP (18) are, in fact, a mixed-integer convex formulation of the feasible set of the bilinear program. In Section 7.2.2 we will explain this result through a simple geometric argument.
- Section 7.3 concerns the strength of our MICP. Using Lemma 1 we have relaxed the constraints corresponding to each vertex in our graph independently (see Remark 9). Ideally, we would like these relaxations to be as tight as possible and coincide with the convex hull of the corresponding bilinear constraints. We will see that this is not the case (if it was, we would have $P = NP$). Compared to an exact characterization of the convex hull, our formulation results in a limited loss of strength but in a considerable reduction in size.
- Finally, in Section 7.4 we will isolate the geometric properties of our problem that in Section 5.3 allowed us to reduce the size of the proposed MICP.

The upcoming analysis will highlight multiple similarities between our convexification method and the RLT [86, 85], which, in turn, has its roots in the Sherali-Adams [84] and the Lovász-Schrijver [57] hierarchies for 0-1 optimization problems. (See also [87] for a concise overview of RLTs and [51] for a comparison of these classical hierarchies.) Except for Section 10, the remaining of this paper does not depend on the technical results presented in this section.

7.1 Generalization of the Mixed-Integer Convex Formulation

Let us restate in a more concise form the bilinear program (10). As in Section 5.2 we stack the auxiliary variables $(z_e)_{e \in E_v^{\text{in}}}$ and $(y_e)_{e \in E_v^{\text{out}}}$ in the matrix M_v which, using the bilinear constraints (10c), is seen

to coincide with the outer product $= x_v \varphi_v^\top$. Defining the nonconvex sets $\Omega_v := \{(\varphi, x, M) : \varphi \in \Phi_v, x \in X_v, M = x \varphi^\top\}$ for all $v \in V$, problem (10) is compactly stated as

$$\text{minimize} \quad \sum_{e \in E} \tilde{\ell}_e(y_e, z_e, \varphi_e) \quad (24a)$$

$$\text{subject to} \quad (\varphi_v, x_v, M_v) \in \Omega_v, \quad \forall v \in V. \quad (24b)$$

In this subsection we retrace the steps from Section 5.2 and we show that Lemma 1 is, in fact, a general-purpose tool to construct convex relaxations Ω'_v for sets of the form of Ω_v . The MICP (18) will be then stated as

$$\text{minimize} \quad \sum_{e \in E} \tilde{\ell}_e(y_e, z_e, \varphi_e) \quad (25a)$$

$$\text{subject to} \quad (\varphi_v, x_v, M_v) \in \Omega'_v, \quad \forall v \in V, \quad (25b)$$

$$\varphi_e \in \{0, 1\}, \quad \forall e \in E, \quad (25c)$$

and multiple properties of this program will be inferred by analyzing the relationship between the sets Ω_v and Ω'_v .

As seen in Section 5.2, valid inequalities play a central role in our convexification method. Let us then start with a formal definition.

Definition 3. For a set $S \subseteq \mathbb{R}^n$, we define the set of *valid linear inequalities* as

$$S^\circ := \{(a, b) \in \mathbb{R}^{n+1} : a^\top x + b \geq 0 \text{ for all } x \in S\}.$$

Using the definition, it is immediately verified that S° is a convex cone, even if S is nonconvex. The next lemma describes the set S and its perspective $\tilde{S}$ in terms of the valid inequalities S° .

Lemma 3. Let $S \subseteq \mathbb{R}^n$ be a closed convex set.

(a) We have $S = \{x : a^\top x + b \geq 0 \text{ for all } (a, b) \in S^\circ\}$.

(b) Assume, in addition, S to be bounded. We have $\tilde{S} = \{(x, \lambda) : a^\top x + b\lambda \geq 0 \text{ for all } (a, b) \in S^\circ\}$.

Note that, essentially, Lemma 3(b) states that, for a compact convex set S , the cones $\tilde{S}$ and S° are dual.

Proof of Lemma 3. Point (a) is verified by checking mutual inclusion: one direction follows from the definition of S° , the contrapositive of the other direction is easily proven using the separating-hyperplane theorem in its strict version (see, e.g., [9, Example 2.20]). Point (b) is simply Lemma 2 applied to the description of S from point (a). (Note that in this description of $\tilde{S}$ the condition $\lambda \geq 0$ needs not to be stated explicitly since $(0, 1) \in S^\circ$.) $\square$

We go back to the analysis of the bilinear program (24). Let us drop the dependence on the vertex v from our notation, and momentarily extend the discussion to two generic compact convex sets $\Phi \subset \mathbb{R}^n$ and $X \subset \mathbb{R}^d$. We consider the following nonconvex set:

$$\Omega := \{(\varphi, x, M) : \varphi \in \Phi, x \in X, M = x \varphi^\top\}. \quad (26)$$

We generate valid inequalities for Ω by multiplying valid inequalities for Φ and X . Let $(a, b) \in \Phi^\circ$ and $(c, d) \in X^\circ$. For any point $(\varphi, x, M) \in \Omega$, we have

$$0 \leq (a^\top \varphi + b)(c^\top x + d) = c^\top M a + d a^\top \varphi + b c^\top x + b d. \quad (27)$$

Since the expression on the right-hand side is linear in φ , x , and M , we have the following lemma.

Lemma 4. *The following set is convex and contains Ω :*

$$\Omega' := \{(\varphi, x, M) : c^\top Ma + da^\top \varphi + bc^\top x + bd \geq 0 \text{ for all } (a, b) \in \Phi^\circ \text{ and } (c, d) \in X^\circ\}. \quad (28)$$

Depending on whether describing the valid inequalities of one of the sets Φ and X is simpler than for the other, the following descriptions of Ω' can be more convenient.

Lemma 5. *The following are equivalent descriptions of the set Ω' :*

$$\left\{ (\varphi, x, M) : \begin{bmatrix} Ma + bx \\ a^\top \varphi + b \end{bmatrix} \in \tilde{X} \text{ for all } (a, b) \in \Phi^\circ \right\}, \quad (29a)$$

$$\left\{ (\varphi, x, M) : \begin{bmatrix} M^\top c + d\varphi \\ c^\top x + d \end{bmatrix} \in \tilde{\Phi} \text{ for all } (c, d) \in X^\circ \right\}. \quad (29b)$$

Proof. Consider the first set. Using Lemma 3(b), we rewrite the condition $(Ma + bx, a^\top \varphi + b) \in \tilde{X}$ as $c^\top (Ma + bx) + d(a^\top \varphi + b) \geq 0$ for all $(c, d) \in X^\circ$. Expanding the products we get back the definition of Ω' in (28). The equivalence of (29b) and (28) is shown the same way. $\square$

We notice the following parallels between Lemma 5 and the results from Section 5.2:

- The description of Ω' in (29a) uses exactly the procedure from Lemma 1(b). There we arrived to this result via a direct argument, i.e., by multiplying in (12) the valid inequalities for Φ_v by the corresponding vertex position x_v . The path taken here is longer, but it shows that the other classes of constraints that we could derive using the same principle, namely (28) and (29b), would not tighten our relaxation.
- In Lemma 1(a) we analyzed the case of a valid linear equality $a^\top \varphi + b = 0$ separately, and we noticed that such a constraint leads to a spatial equality constraint $Ma + bx = 0$. This distinction was done for clarity, but it is unnecessary. Rewriting the valid equality as two inequalities $(a, b) \in \Phi^\circ$ and $-(a, b) \in \Phi^\circ$, we notice that among the conditions in (29a) we have both $(Ma + bx, a^\top \varphi + b) \in \tilde{X}$ and $-(Ma + bx, a^\top \varphi + b) \in \tilde{X}$. Since $\tilde{X}$ is a pointed cone, these imply $(Ma + bx, a^\top \varphi + b) = 0$ as in (13).

Remark 12. The step in (27), of multiplying linear inequalities on the decision variables φ and x to generate linear inequalities on the product variables M , is a common paradigm in optimization; it represents the building block of RLTs [86, 85]. It is Lemma 5 that differentiates our approach from these classical techniques: the perspective operations in (29) allow us to treat one of the two convex sets Φ and X as a black box, and to summarize all the valid inequalities for this set via a convex cone constraint.

7.2 The Case of a Polytopic Set Φ

We restrict the analysis to a polytopic set Φ , and we illustrate two important properties that our convexification technique enjoys under this assumption. Note that, given the symmetric roles played by the sets Φ and X , results analogous to the ones below hold in case of a polytopic X .

7.2.1 Finitely-Generated Convex Relaxations Ω'

Though convex, the relaxation Ω' is inoperable in its current form since, in each of its descriptions, it involves infinitely many constraints. As one might expect, the first advantage of working with a polytopic set Φ is that only a finite subset of the conditions in (29a) is actually needed. We already took advantage of this observation in Section 5.2 where we applied Lemma 1 only to the constraints defining a facet of Φ_v . Our next step is to formally show that this omission was lossless.

Let Φ be a polytope with halfspace representation $\{\varphi : a_i^\top \varphi + b_i \geq 0 \text{ for all } i \in I\}$, where I is a finite index set. We claim that the following is an equivalent description of the convex set Ω' :

$$\Omega' = \left\{ (\varphi, x, M) : \begin{bmatrix} Ma_i + b_i x \\ a_i^\top \varphi + b_i \end{bmatrix} \in \tilde{X} \text{ for all } i \in I \right\}. \quad (30)$$

Here the conditions in (29a) are only enforced for the inequalities $(a_i, b_i) \in \Phi^\circ$ defining the polytope Φ . The next proposition proves this claim by showing that redundant inequalities for the set Φ are mapped to redundant constraints in (29a).

Lemma 6. *Let $(a, b) \in \Phi^\circ$. There exist scalars $\alpha_i \geq 0$ such that $a = \sum_{i \in I} \alpha_i a_i$ and $b \geq \sum_{i \in I} \alpha_i b_i$.*

Proof. Since $(a, b) \in \Phi^\circ$, the minimum of $a^\top \varphi + b$ over all $\varphi \in \Phi$ is nonnegative. The dual of this minimization requires maximizing $b - \sum_{i \in I} \alpha_i b_i$ over all nonnegative α_i such that $a = \sum_{i \in I} \alpha_i a_i$. The thesis follows from LP strong duality. $\square$

Proposition 2. *Let $(a, b) \in \Phi^\circ$. The constraint $(Ma + bx, a^\top \varphi + b) \in \tilde{X}$ is redundant for the set on the right-hand side in (30).*

Proof. Let (φ, x, M) be a point in the set from (30). Define the scalars α_i as in Lemma 6 and let $\beta := b - \sum_{i \in I} \alpha_i b_i \geq 0$. We take a linear combination with coefficients α_i of the conditions in (30). On the left-hand side we get the vector $\sum_{i \in I} \alpha_i (Ma_i + b_i x, a_i^\top \varphi + b_i) = (Ma + (b - \beta)x, a^\top \varphi + b - \beta)$. On the right-hand side, recalling that $\tilde{X}$ is a convex cone, we obtain the set $\sum_{i \in I} (\alpha_i \tilde{X}) = \sum_{i \in I} \tilde{X} = \tilde{X}$. The thesis is verified by summing the inclusion we just derived with $\beta(x, 1) \in \tilde{X}$. $\square$

Coming back to the SPP, the last proposition certifies that a redundant linear inequality for the polytope Φ_v cannot be used to strengthen our MICP via Lemma 1. On the other hand, as long as we agree not to leverage couplings between nonincident flows as mentioned in Remark 9, there are no further linear inequalities that we can add to our program to cut a portion of Φ_v either. In fact, as seen in Section 4.1 any such inequality would cut out a potentially-optimal integer flow φ_v . We conclude that our use of Lemma 1 in Section 5.2 has been exhaustive.

Remark 13. Proposition 2 gives us a simple proof of the claim made in Remark 8 by which the convex constraints (18b)–(18d) imply the condition $x_v \in X_v$. Specializing Proposition 2 to the valid inequality $(0, 1) \in \Phi^\circ$, we see that the constraint $(x, 1) \in \tilde{X}$, which is equivalent to $x \in X$, is verified by any point in Ω' .

7.2.2 Tightness of the Envelope Ω' at the Extreme Points of Φ

Narrowing the class of convex sets Φ to polytopes has allowed us to describe the convex relaxation Ω' through a finite number of conditions in (30). There is another, less obvious, advantage of working with a polytopic set Φ that yields a simpler and more insightful argument for the validity of our MICP formulation in Theorem 3. Let us analyze the tightness of the inclusion $\Omega \subseteq \Omega'$, zooming in on any of the extreme points $\text{ext}(\Phi)$ of the polytope Φ .

Lemma 7. For any extreme point $\hat{\varphi} \in \text{ext}(\Phi)$, define $L := \{(\varphi, x, M) : \varphi = \hat{\varphi}\}$. We have $\Omega \cap L = \Omega' \cap L$.

Proof. The inclusion $\Omega \cap L \subseteq \Omega' \cap L$ follows from $\Omega \subseteq \Omega'$. Let x and M be such that $(\hat{\varphi}, x, M) \in \Omega'$, to prove the reverse inclusion we need to show that $M = x\hat{\varphi}^\top$. Since $\hat{\varphi}$ is an extreme point of Φ , we can always identify n linearly-independent inequalities defining Φ that are active at $\hat{\varphi}$. Let $\hat{A} \in \mathbb{R}^{n \times n}$ be the invertible matrix whose rows are the vectors a_i corresponding to these active inequalities, and let the vector $\hat{b} \in \mathbb{R}^n$ be constructed analogously from the scalars b_i . We have $\hat{\varphi} = -\hat{A}^{-1}\hat{b}$. Evaluated at $\varphi = \hat{\varphi}$, the spatial constraints in (30) corresponding to these inequalities read $Ma_i + b_i x \in (a_i^\top \hat{\varphi} + b_i)X = \{0\}$, and can be stacked in the matrix equality $M\hat{A}^\top + xb^\top = 0$. We conclude that $M = -xb^\top \hat{A}^{-\top} = -x(\hat{A}^{-1}b)^\top = x\hat{\varphi}^\top$. $\square$

In words, this lemma states that, if we restrict our attention to the extreme points of Φ , the convex relaxation Ω' of the set Ω is perfectly tight. Having noticed this, the validity of our MICP formulation is immediately established.

Alternative proof of Theorem 3. The constraints of the MICP (25) enforce both $\varphi_v \in \Phi_v$ and $\varphi_v \in \{0, 1\}^{|E_v|}$. As seen in Remark 6, the flows in $\Phi_v \cap \{0, 1\}^{|E_v|}$ are the extreme points $\text{ext}(\Phi_v)$ of the local flow polytope. Thus, by Lemma 7, the constraint sets Ω_v and Ω'_v are interchangeable in (25b). The feasible set of the bilinear program (24), subject to the extra constraints $\varphi_e \in \{0, 1\}$ for all $e \in E$, is then equal to the feasible set of the MICP (25). $\square$

7.3 Tightness of the Convex Relaxation

In this subsection we analyze the tightness of the inclusion $\text{conv}(\Omega) \subseteq \Omega'$. Ideally, we would like our convex relaxation to be as tight as possible and this inclusion to be an equality. We will see that, for generic convex sets Φ and X , this is not the case (if it was, we would have $P = NP$). In the special case of a polytopic set Φ , we will see that an explicit description of $\text{conv}(\Omega)$ can be derived using disjunctive-programming techniques, and that the size of this description is proportional to the number of extreme points of Φ . Since the local flow polytopes Φ_v in our SPP have a small number of extreme points, $|\text{ext}(\Phi_v)| = O(|E_v^{\text{in}}||E_v^{\text{out}}|)$, this will yield a tractable description of the sets $\text{conv}(\Omega_v)$. However, this description will be substantially larger than the one of Ω'_v and, in practice, its higher strength will not be worth the increase in size.

Let us start by illustrating a simple example where the equality $\text{conv}(\Omega) = \Omega'$ does hold.

Example 4. Let Φ and X be intervals in the real line: $\Phi := [\varphi_{\min}, \varphi_{\max}]$ and $X := [x_{\min}, x_{\max}]$. A halfspace representation of Φ is given by $a_1 := 1$, $a_2 := -1$, $b_1 := -\varphi_{\min}$, $b_2 := \varphi_{\max}$, and $I := \{1, 2\}$. Expanding the conditions in (30), we get the following description of Ω' :

$$\varphi \geq \varphi_{\min}, \quad M - \varphi_{\min}x \in (\varphi - \varphi_{\min})[x_{\min}, x_{\max}], \quad (31a)$$

$$\varphi \leq \varphi_{\max}, \quad \varphi_{\max}x - M \in (\varphi_{\max} - \varphi)[x_{\min}, x_{\max}]. \quad (31b)$$

These linear inequalities are recognized to define the classical McCormick envelope [65] of the bilinear surface $M = \varphi x$, which is depicted in Figure 2 for $\varphi_{\min} = 0$ and $\varphi_{\max} = x_{\max} = -x_{\min} = 1$. The McCormick envelope is known to coincide with the convex hull of this surface: thus, in this case, we have $\Omega' = \text{conv}(\Omega)$.

As a side note, notice that Lemma 7 applies to the case depicted in Figure 2: at the extreme points $\text{ext}(\Phi) = \{0, 1\}$ of Φ the envelope Ω' adheres perfectly to Ω .

The next example shows that, in general, the containment of $\text{conv}(\Omega)$ in Ω' can be loose.

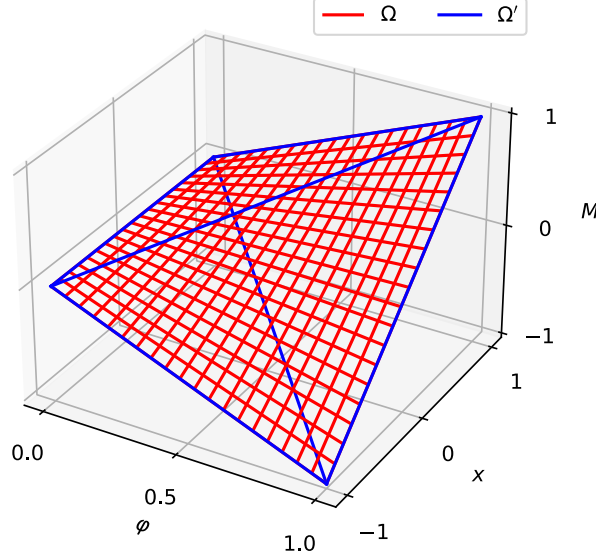


Figure 2: The nonconvex set Ω and its convex relaxation Ω' from Example 4, depicted for $\varphi_{\min} = 0$ and $\varphi_{\max} = x_{\max} = -x_{\min} = 1$. In the simple case where the sets Φ and X are intervals in the real line, the set Ω' coincides with the classical McCormick envelope [65], i.e., the convex hull of Ω .

Example 5. Let $\Phi := X := [-1, 1]^2$. Consider the problem of maximizing $M_{11} + M_{12} + M_{21} - M_{22}$ subject to the constraint $(\varphi, x, M) \in \Omega$. The optimal value of this program is 2, and it is achieved for

$$\varphi = x = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \quad M = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}.$$

Since the objective is linear, by substituting the constraint set Ω with its convex hull we do not alter the optimal value. On the other hand, substituting Ω with its convex relaxation Ω' the optimal value increases to 4, with maximizers

$$\varphi = x = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \quad M = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}.$$

Therefore, in this case, the convex hull of Ω is strictly smaller than Ω' .

Remark 14. The existence of a counterexample as the one just illustrated should not surprise. In fact, any bilinear program of the form

$$\begin{aligned} & \text{maximize} && p^\top \varphi + q^\top x + \varphi^\top R x \\ & \text{subject to} && \varphi \in \Phi, \, x \in X, \end{aligned}$$

with Φ and X convex, is easily rewritten as the maximization of a linear function of (φ, x, M) over the nonconvex set Ω . Optimizations of this kind have been deeply studied in the global optimization literature (see [45] Sections I.2.4 and IX.1] and the references therein) and they belong to the class of NP-hard problems. Since, for tractable convex sets Φ and X , optimizing a linear function over Ω' takes polynomial time, the equality $\Omega' = \text{conv}(\Omega)$ would imply $P = NP$.

As in Section 7.2 in the last part of this subsection we focus on the case of a polytopic set Φ . The following proposition shows that the shape of the convex hull of Ω is affected only by the points in Ω that correspond to an extreme point of Φ .

Proposition 3. We have $\text{conv}(\Omega) = \text{conv}(\Omega \cap C)$, where $C := \{(\varphi, x, M) : \varphi \in \text{ext}(\Phi)\}$.

Proof. See Appendix B.3. □

Proposition 3 allows us to derive an actionable description of the set $\text{conv}(\Omega)$. In fact, the set $\Omega \cap C$ can be expressed as the union of a convex set per extreme point of Φ ,

$$\Omega \cap C = \bigcup_{\hat{\varphi} \in \text{ext}(\Phi)} \{\hat{\varphi}\} \times \{(x, M) : x \in X, M = x\hat{\varphi}^\top\},$$

and the convex hull of this *disjunctive convex set* can be explicitly described using disjunctive-programming techniques [13, 3]. The resulting representation of $\text{conv}(\Omega)$ involves a number of convex conditions proportional to $|\text{ext}(\Phi)|$. In general, this number can be exponential in the dimension n of space where Φ lives, however, as seen in Section 4.1 our local flow polytopes Φ_v have a small number of extreme points that is (at most) bilinear in the indegree $|E_v^{\text{in}}|$ and the outdegree $|E_v^{\text{out}}|$ of vertex v . Therefore, for our SPP, the route just described leads to a polynomial-size MICP that is stronger than our formulation (21). This MICP is described in detail in Appendix A.2 and tested numerically in Appendix A.3. Here we only mention that, though polynomial, in our numerical experience, the size of this MICP is generally prohibitive. With a little sacrifice in strength, the MICP (21) scales linearly with the size of the graph G and performs much better in practice.

7.4 Orthogonal Projection of the Set Ω'

The last part of this analysis is devoted to the reduced MICP (21). In Section 5.3 we observed that each vertex position x_v appears in only one of the constraints of the MICP (18). This allowed us to drop the decision variables x_v from our program, together with all the constraints involving them. Geometrically, this operation corresponds to projecting the feasible set of the MICP (18) onto the space of the variables $\{\varphi_e, y_e, z_e\}_{e \in E}$. With the notation of this section, the reduced MICP takes then the form

$$\text{minimize} \quad \sum_{e \in E} \tilde{\ell}_e(y_e, z_e, \varphi_e) \tag{32a}$$

$$\text{subject to} \quad (\varphi_v, M_v) \in \text{proj}_{(\varphi, M)}(\Omega'_v), \quad \forall v \in V, \tag{32b}$$

$$\varphi_e \in \{0, 1\}, \quad \forall e \in E, \tag{32c}$$

where the vertex positions x_v are projected out the constraint sets Ω'_v . Problem (32) is immediately verified to be equivalent to (25), since our objective function does not depend on the variables x_v .

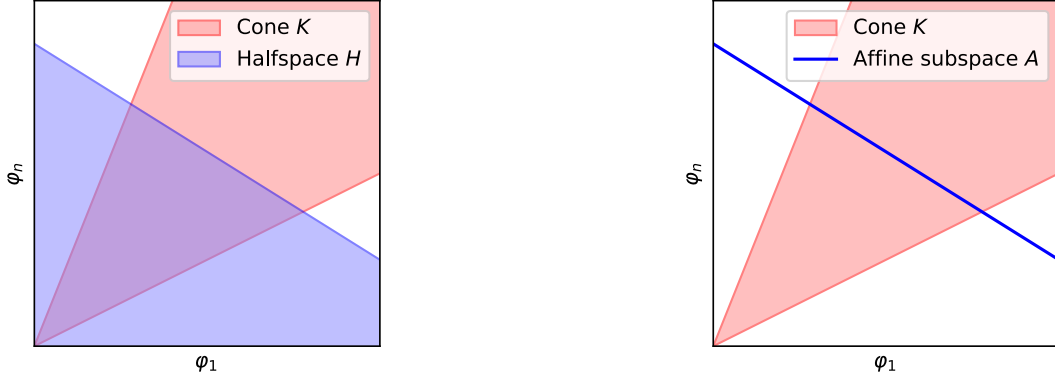
In general, projecting the set Ω' is a complicated operation: in case of polytopic sets Φ and X , for example, the set Ω' is also a polytope and its projection can be delimited by a number of halfspaces exponential in d . However, sets Φ with particular geometry can make this operation very simple.

Let us consider the case depicted in Figure 3a, where Φ is the intersection of a convex cone K and a halfspace $H := \{\varphi : a^\top \varphi + 1 \geq 0\}$. For simplicity, let us also assume K to be polyhedral: $K := \{\varphi : a_i^\top \varphi \geq 0 \text{ for all } i \in I\}$. In this case, the constraints defining the convex set Ω' in (30) are

$$(Ma_i, a_i^\top \varphi) \in \tilde{X}, \quad \forall i \in I, \tag{33a}$$

$$(Ma + x, a^\top \varphi + 1) \in \tilde{X}. \tag{33b}$$

Among these conditions, only the spatial constraint $Ma + x \in (a^\top \varphi + 1)X$ in (33b) involves the variable x . Removing this from our description of Ω' , we get an explicit representation of the set $\text{proj}_{(\varphi, M)}(\Omega')$.



(a) Intersection of a cone K and a halfspace H .

(b) Intersection of a cone K and an affine subspace A .

Figure 3: The two kinds of polytopes Φ for which we compute the orthogonal projection of set Ω' onto the space of the variables φ and M .

To get a lifting function that maps any point in the projection to a point in Ω' , we simply define x so that the constraint we removed is always verified:

$$x := -Ma + (a^\top \varphi + 1)\theta, \quad (34)$$

where θ is any point in X . Notice that in our SPP the polytopes Φ_v have this structure for all $v \notin \{s, t\}$: the only nonhomogeneous constraint is the degree constraint (4c) and, representing it through the vector $a := (0_{|E_v^{\text{in}}|}, -1_{|E_v^{\text{out}}|})$, the lifting formula (34) gives us exactly (19).

The analysis does not change if, as in Figure 3b we intersect the cone K with an affine subspace $A := \{\varphi : a^\top \varphi + 1 = 0\}$. The spatial constraint in (33b) becomes now $Ma + x = 0$. Dropping this equality we obtain an explicit description of $\text{proj}_{(\varphi, M)}(\Omega')$, while the lifting

$$x := -Ma \quad (35)$$

maps any point in the set $\text{proj}_{(\varphi, M)}(\Omega')$ to a point in Ω' . In our SPP, the polytope Φ_t has this structure, with the conservation of flow (4b) being the nonhomogeneous constraint. After a small rearrangement of the equations governing its flow, the same holds for the source polytope Φ_s . The lifting formulas in (20) are obtained from (35) by letting $a := (0_{|E_s^{\text{out}}|}, -1_{|E_s^{\text{in}}|})$ and $a := (-1_{|E_t^{\text{in}}|}, 0_{|E_t^{\text{out}}|})$.

Example 6. We continue with Example 4 using the numeric values from Figure 2. In this case the polytope Φ falls into the first category analyzed above, with the halfspace H being defined by $a := -1$, and the cone K by $a_1 := 1$ and $I := \{1\}$. The conditions (31a) and (31b) defining the McCormick envelope correspond to (33a) and (33b), respectively. The projection of Ω' is then obtained by dropping spatial constraint in (31b) from the description of the envelope: $\text{proj}_{(\varphi, M)}(\Omega') = \{(\varphi, M) : \varphi \in [0, 1], M \in \varphi[-1, 1]\}$. The correctness of this expression can be visually confirmed from Figure 2.

8 Dual Optimization Problem

In this section we analyze the dual of the convex relaxation of the MICP (21). Additional parallels between this program and the LP formulation (5) of the classical SPP are drawn in Sections 8.1 and 8.2. An informative lower bound on the optimal value of our convex relaxation is derived using duality in Section 8.3.

8.1 Dual of the Classical Shortest-Path Linear Program

As a reference for the upcoming analysis, the dual of the LP formulation (5) of the classical SPP is

$$\text{maximize } p_s - p_t \quad (36a)$$

$$\text{subject to } p_u - p_v \leq c_e, \quad \forall e = (u, v) \in E, \quad (36b)$$

where p_v denotes the multiplier of the conservation of flow (4b) at vertex v .⁸ These multipliers are well-known to be interpretable as potentials: the objective asks to maximize the potential jump between the source s and the target t , the constraint ensures that the potential jump along each edge does not exceed the length of the edge itself.

8.2 Dual of the Proposed Mixed-Integer Convex Program

Instead of deriving the dual of the convex relaxation of the MICP (21), we state it directly and we prove weak duality (i.e. that the optimal value of the dual problem bounds from below the one of the primal). In fact, this is the only property of the dual program that we use in this paper.

We let the following Lagrange multipliers be the decision variables of the dual program. For each edge $e \in E$, we pair the first nonnegativity constraint in (21b) with the dual variables $a_e \in \mathbb{R}^d$ and $b_e \in \mathbb{R}$, and the second with $\alpha_e \in \mathbb{R}^d$ and $\beta_e \in \mathbb{R}$. For all $v \in V$, we associate $p_v, q_v \in \mathbb{R}$ to the conservation of flow and the degree constraint in (21c), respectively. Finally, for $v \in V - \{s, t\}$, we let $r_v \in \mathbb{R}^d$ be the multiplier for the spatial conservation of flow (21d). The dual of the convex relaxation of the MICP (21) is

$$\text{maximize } p_s - p_t - \sum_{v \in V - \{t\}} q_v \quad (37a)$$

$$\text{subject to } p_u - p_v - q_u + b_e + \beta_e \leq -\ell_e^*(r_u + a_e, -r_v + \alpha_e), \quad \forall e = (u, v) \in E, \quad (37b)$$

$$(a_e, b_e) \in X_u^\circ, (\alpha_e, \beta_e) \in X_v^\circ, \quad \forall e = (u, v) \in E, \quad (37c)$$

$$q_v \geq 0, \quad \forall v \in V, \quad (37d)$$

$$r_s = r_t = 0. \quad (37e)$$

Here ℓ_e^* denotes the conjugate function⁹ of ℓ_e and X_v° is the set of valid linear inequalities for the set X_v (as in Definition 3). Note also that r_s and r_t are auxiliary decision variables whose only role is to simplify the final expression of the dual problem.

The following proposition shows that weak duality holds for the pair of optimization problems under analysis.

Proposition 4. *The optimal value of program (37) bounds from below the optimal value of the convex relaxation of the MICP (21).*

Proof. See appendix B.4 □

Let us briefly comment on the structure of the dual program (37). Comparing it with the LP (36), we notice that the objective (37a) still maximizes the potential jump $p_s - p_t$, but it also includes

⁸As discussed in Remark 5, the degree constraints (4c) are redundant for this LP. Hence their multipliers do not appear in the dual program (36).

⁹For a function $f : \mathbb{R}^n \rightarrow \mathbb{R} \cup \{\infty\}$, the *conjugate function* is defined as $f^*(a) := \sup_x (a^\top x - f(x))$. Note that, since f^* is the pointwise supremum of affine functions, it is always convex, even when f is not.

the multipliers of the degree constraints (which are redundant in the classical SPP). Constraint (37b) clearly resembles (36b): the potential jump $p_u - p_v$, together with an extra term, is upper bounded by a concave expression associated to the edge length. The connection is even more evident noticing that, for $\ell_e(x_u, x_v) := c_e$, we have $\ell_e^* = -c_e$ (together with the constraints $r_u = -a_e$ and $r_v = \alpha_e$) and the right-hand sides of (37b) and (36b) coincide. The constraint (37c), on the other hand, is purely due to the spatial nature of our problem, as its primal counterpart (21b).

Problem (37) is always feasible and its cost is nonnegative. This is seen by setting all the multipliers to zero: all the constraints are verified and the cost is zero.

8.3 A Simple Lower Bound on the Optimal Cost

We now take advantage of the dual program (37) to perform a brief “sanity check” on the strength of our MICP (21). The reduction argument from Theorem 1 suggests a simple lower bound on the optimal cost of the SPP (1), here we show that the convex relaxation of our MICP always recovers this bound.

Consider a problem setup in which all the edges $e = (u, v) \in E$ share the same length function

$$\ell_e(x_u, x_v) := \ell(x_v - x_u), \quad (38)$$

which only depends on the distance $x_v - x_u$, and not on x_u and x_v independently. Assume also that $\ell(0) = 0$. Momentarily, we focus our attention on the case in which the source and target sets are single points: $X_s := \{\theta_s\}$ and $X_t := \{\theta_t\}$. With the goal of lower bounding the optimal cost of the SPP (1), we can drop the constraints $x_v \in X_v$ for all $v \in V - \{s, t\}$. Similarly to the proof of Theorem 1, an optimal solution for this relaxed problem is obtained by first detecting an s - t path $\pi = (v_k)_{k=0}^K$ with maximum number K of edges, and then by arranging the points x_{v_k} at equal distance along the line segment connecting θ_s and θ_t . The cost of this arrangement is

$$K\ell\left(\frac{\theta_t - \theta_s}{K}\right) =: \tilde{\ell}(\theta_t - \theta_s, K). \quad (39)$$

We cannot ask the convex relaxation of our MICP (21) to always recover this value: as discussed in Section 3 this would yield a polynomial-time algorithm for the HPP, which is NP-complete. On the other hand, any s - t path can contain at most $K = |V| - 1$ edges, therefore a simple lower bound on the optimal cost of our SPP is $\tilde{\ell}(\theta_t - \theta_s, |V| - 1)$.

More generally, when the sets X_s and X_t are full dimensional, we have the following result.

Proposition 5. *Assume all the edges to share the common length function (38). The optimal cost of the convex relaxation of the MICP (21) is not smaller than the optimal cost of*

$$\text{minimize} \quad \tilde{\ell}(x_t - x_s, |V| - 1) \quad (40a)$$

$$\text{subject to} \quad x_s \in X_s, \quad x_t \in X_t. \quad (40b)$$

Proof. See Appendix B.5. □

The proof of this proposition requires an intricate interplay between the dual variables in (37): this shows that all the constraints in (21) are essential for our formulation to pass this sanity check. In particular, contrarily to what happens for the LP (5) (see Remark 5), the degree constraint in (21c) is not redundant for our MICP.

9 Application to Optimal Control of Hybrid Dynamical Systems

We now show how optimal-control problems for discrete-time hybrid dynamical systems can be formulated as the SPP presented in Section 2. We focus on the broad class of PieceWise-Affine (PWA) systems [88], and we analyze both the cases in which the time horizon of the control problem is to be optimized or it is fixed a priori.

A PWA system has the structure

$$\zeta(k+1) = A_i \zeta(k) + B_i w(k) + c_i \quad \text{if } (\zeta(k), w(k)) \in D_i, \quad (41)$$

where $k \in \mathbb{Z}$ is the discrete time, $\zeta \in \mathbb{R}^{d_\zeta}$ is the system state, $w \in \mathbb{R}^{d_w}$ is the control input, and i is the system mode which takes values in a finite index set I . In words, we have a collection of $|I|$ affine dynamics $(A_i, B_i, c_i)_{i \in I}$, each of which applies in a different (convex compact) portion D_i of the state and control space.

Loosely speaking, almost any system whose nonlinearity is exclusively due to discrete logics can be written in PWA form [42]. In addition, smooth nonlinear dynamics can be approximated arbitrarily well by using a PWA model [88]. PWA systems have seen a multitude of applications: automotive [8], power electronics [37], robotics [60, 41], and many more [11]. Also a linear system that navigates through an environment with obstacles can be seen as a PWA system, where the convex sets D_i represent regions of collision-free space.

9.1 Problems with Free Time Horizon

Given the initial state $\zeta(1)$, our goal is to find a control sequence $(w(k))_{k=1}^{K-1}$ that drives the system final state $\zeta(K)$ to a given target set $Z \subset \mathbb{R}^{d_\zeta}$, subject to the PWA dynamics (41) and while optimizing some function of the states $\zeta(k)$ and the controls $w(k)$. The time K at which the system reaches the target is not specified a priori.

To transcribe this problem as an SPP [1] we proceed as in Figure 4a. We let the vertices of our graph be $V := \{s, t\} \cup I$: the source s and the target t will be used to enforce the initial and final conditions, while a visit to vertex $i \in I$ will represent a time step spent in region D_i . We let the sets X_v live in the state and control space $\mathbb{R}^d$ with $d := d_\zeta + d_w$, and we denote with $x_v := (\zeta_v, w_v) \in X_v$ the position of vertex v . We pair the source s with the set $X_s := \{\zeta(1)\} \times \{0_{d_w}\}$, the target t with $X_t := Z \times \{0_{d_w}\}$, and the remaining vertices $i \in I$ with the corresponding set D_i . The source s is connected by an edge (s, i) to each of the vertices $i \in I$; similarly we also have the edges (i, t) for all $i \in I$. We might then allow transitions between any pair of distinct vertices in I , in which case we proceed as in Figure 4a and we add all the edges $(i, j) \in I^2$ with $i \neq j$ to our edge set E . Otherwise, we can prevent an undesired mode transition by excluding the corresponding edge. To allow the system to spend multiple time steps in the same region D_i we can proceed as in Remark 2 (not shown in Figure 4a).

Notice that the mode i in which the PWA system (41) is at time $k = 1$ is not uniquely determined by the initial state $\zeta(1)$, but it depends also on the first control $w(1)$. This is the reason why we introduce the fictitious vertex s : defining the length of each edge $e = (s, i)$ as

$$\ell_e(x_s, x_i) := \begin{cases} 0 & \text{if } \zeta_i = \zeta_s \\ \infty & \text{otherwise} \end{cases}, \quad (42)$$

we allow the optimal path π to connect at zero cost the source s to any vertex $i \in I$ for which there exists a w such that $(\zeta(1), w) \in D_i$. For all the remaining edges $e = (i, v)$, where either $v \in I$ or

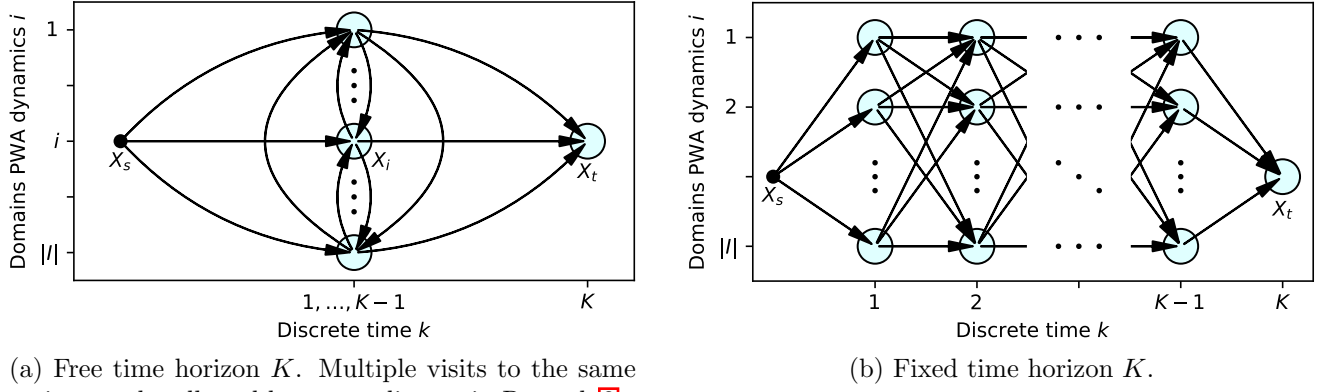


Figure 4: Graphs G for the optimal control of a PWA system in case of free and fixed time horizon K . In both cases, transitions are allowed between any pair of modes.

$v = t$, we define ℓ_e so that state transitions that do not agree with the dynamics incur infinite cost:

$$\ell_e(x_i, x_v) := \begin{cases} \ell'_e(x_i, x_v) & \text{if } \zeta_v = A_i \zeta_i + B_i w_i + c_i, \\ \infty & \text{otherwise} \end{cases}, \quad (43)$$

where ℓ'_e is a suitable cost function. To derive the perspectives of the edge lengths (42) and (43) we can proceed as in Section 6.1.4

For what concerns the transition cost ℓ'_e , we can, e.g., define it as $\ell'_e(x_i, x_v) := 1$. This yields a minimum-time problem where the goal is to reach the target set Z as soon as possible. Otherwise, a popular class of cost functions are positive semidefinite quadratic forms

$$\sum_{k=1}^{K-1} \left(\zeta^\top(k) Q \zeta(k) + w^\top(k) R w(k) \right).$$

This objective balances the magnitude of the control effort and the distance of the system state from the origin. In this case, we let $\ell'_e(x_i, x_v) := \zeta_i^\top Q \zeta_i + w_i^\top R w_i$. When working with finite-horizon problems, it is also frequently useful to enforce a penalty on the magnitude of the terminal state $\zeta(K)$, e.g., $\zeta^\top(K) P \zeta(K)$ with P positive semidefinite. This is a fundamental tool to ensure closed-loop stability of the control system [64]. In our construction, this can be easily achieved by adding to the lengths ℓ'_e of the edges $e = (i, t)$ the term $\zeta_t^\top P \zeta_t$.

After having solved the MICP (21), and recovered the shortest path $\pi = (v_k)_{k=0}^K$, the optimal control sequence is $w(k) = w_{v_k}$ for $k = 1, \dots, K-1$ and the corresponding state trajectory is $\zeta(k) = \zeta_{v_k}$ for $k = 1, \dots, K$.

9.2 Problems with fixed time horizon

Choosing a fixed time horizon K for a control problem is a tricky compromise between performance and computational efficiency. Furthermore, letting K be a decision variable has also several technical advantages [66, 83, 80]. Nevertheless, in some cases, we might need the value of K to be fixed. With some extra effort, also fixed-horizon control problems can be formulated as the SPP (1).

We proceed as in Figure 4b. We include in the vertex set V : the source s , the target t , and a vertex (k, i) for each time step $k = 1, \dots, K-1$ and each mode $i \in I$ of the PWA system. The source X_s

and target X_t sets are defined as before, while we pair each vertex (k, i) with a copy of the convex set D_i . The source s is connected by an edge to each vertex $(1, i)$ for $i \in I$ in the first layer. Similarly, the vertices $(K - 1, i)$ are connected to the target t for all $i \in I$. For $k = 1, \dots, K - 2$, we have an edge from vertex $u = (k, i)$ to vertex $v = (k + 1, j)$ for all $(i, j) \in I^2$. This ensures that every state transition increases the time count by exactly one unit. Since any s - t path in the graph we constructed has exactly K edges, the time available to reach the target is fixed and equal to $K - 1$. For the remaining components of problem (1), the discussion from the previous subsection carries over without any major modification.

Remark 15. Overall, the size of the SPP we construct scales linearly with the time horizon K and quadratically with the number $|I|$ of modes. Conversely, classical formulations for these problems have size linear in both K and $|I|$ [61]. However, as we will see in Section 11.3 the higher strength of our approach is generally worth this price.

10 Extension of Other Classical Graph Problems

As discussed in Section 1.1.1 multiple graph problems with neighborhoods have been studied in the literature, and exact solution algorithms for these problems typically require using very expensive MINCP techniques. Examples are the TSP [36] and the MSTP [6]. Here we show that, under standard convexity assumptions [36, 6], the techniques we described in Section 7 can also be applied to derive strong MICP formulations of these problems. This novel perspective leads to substantially easier optimization problems, and it has the potential to strongly outperform existing formulations. A thorough numerical comparison of these formulations is under completion.

Given a graph $G := (V, E)$, which we assume for simplicity to be directed, many combinatorial optimization problems require identifying a subset E^* of the edge set E that verifies given feasibility conditions and which is optimal according to some criterion. Typically, these are formulated as an Integer Linear Program (ILP) of the form

$$\text{minimize} \quad \sum_{e \in E} c_e \varphi_e \tag{44a}$$

$$\text{subject to} \quad \varphi \in \Phi \cap \{0, 1\}^{|E|}, \tag{44b}$$

where $\varphi := (\varphi_e)_{e \in E}$. The edge set E^* is parameterized by the binary variables φ_e as $E^* = \{e \in E : \varphi_e = 1\}$, the polytope Φ embodies the feasibility conditions, and the optimality criterion is described by a linear function of the variables φ , which assigns a finite nonnegative cost c_e to each edge $e \in E$.

We extend the graph problem modeled by the ILP (44) to its version in graphs of convex sets (or with neighborhoods) as done for the SPP. We let the position x_v of vertex $v \in V$ be a decision variable, constrained in the set X_v , and we let the length of the edge $e = (u, v) \in E$ be $\ell_e(x_u, x_v)$. The sets X_v and the functions ℓ_e are subject to the assumptions listed in Section 2. To formulate this graph problem as an MICP, we follow the steps in Section 5.1. We define the auxiliary variables $y_e := \varphi_e x_u$ and $z_e := \varphi_e x_v$ for each edge $e = (u, v) \in E$, and we express our generalized problem as a mixed-integer

program with bilinear constraints:

$$\text{minimize } \sum_{e \in E} \tilde{\ell}_e(y_e, z_e, \varphi_e) \quad (45a)$$

$$\text{subject to } x_v \in X_v, \quad \forall v \in V, \quad (45b)$$

$$y_e = \varphi_e x_u, \quad z_e = \varphi_e x_v, \quad \forall e = (u, v) \in E, \quad (45c)$$

$$\varphi \in \Phi \cap \{0, 1\}^{|E|}. \quad (45d)$$

Arrived at this point, in the case of the SPP, we used the fact that the flow constraint $\varphi_v \in \Phi_v$ in (10b) couples only the flow variables incident with vertex v . This very convenient structure of the SPP allowed us to express the constraints in problem (10) in terms of the nonconvex sets Ω_v as in Section 7.1 and then to convexify the problem as in (25). On the other hand, condition (45d) might involve coupling constraints between flows of nonincident edges (e.g., the subtour-elimination constraints for the TSP by Dantzig, Fulkerson, and Johnson [18]). There are two way in which this issue can be addressed.

One option is just to separate the constraints that involve flows with a common vertex from the ones that couple nonincident flows. Including only the first in the description of the polytopes Φ_v , we can then proceed as for the SPP and derive a convex constraint of the form (13) or (14) for each constraint in the first group. Assuming without loss of generality that $\Phi_v \subseteq [0, 1]^{|E_v|}$, the MICP we get is a valid problem formulation: in fact, this assumption ensures that any point $\varphi_v \in \Phi_v \cap \{0, 1\}^{|E_v|}$ is an extreme point of Φ_v and, by Lemma 7, our convex relaxation is exact in these points. The formulation resulting from this approach is compact but it might be weak.

Similarly to Remark 9 the second option is to introduce extra auxiliary variables that represent the product of each flow φ_e and each vertex position x_v , even if edge e is not incident with vertex v . This gives us a total of $d|V||E|$ auxiliary continuous variables $M = x\varphi^\top$, where the vector $x := (x_v)_{v \in V}$ lives in the Cartesian product $X := \prod_{v \in V} X_v$. Defining the set Ω as in (26), problem (45) becomes

$$\text{minimize } \sum_{e \in E} \tilde{\ell}_e(y_e, z_e, \varphi_e) \quad (46a)$$

$$\text{subject to } (\varphi, x, M) \in \Omega, \quad (46b)$$

$$\varphi \in \{0, 1\}^{|E|}, \quad (46c)$$

where the vectors y_e and z_e can be selected from the entries of M . To obtain an MICP formulation of this mixed-integer nonconvex program, we relax constraint (46b) as $(\varphi, x, M) \in \Omega'$, where the set Ω' is defined as in (30). The validity of the resulting formulation is again ensured under the assumption $\Phi \subseteq [0, 1]^{|E|}$. This second option yields larger but stronger optimization problems.

Remark 16. In some cases, an explicit description of the halfspaces defining the polytope Φ might not be available, and the set Φ might be given as the projection onto the space of the variables φ of a higher-dimensional polytope. Because of the need of auxiliary projection variables, in these cases problem (44) is called an *extended formulation* [17]. Examples are the Miller, Tucker, and Zemlin formulation of the TSP [67], or the MSTP formulation from [17 Section 6.1]. In these cases, we have a decision identical to the one just described: either we exclude the constraints involving the projection variables from the convexification process, or we include them at the price of introducing extra variables that represent the product of the projection variables and the vertex positions x_v . The first route yields a smaller formulation, the second a stronger one.

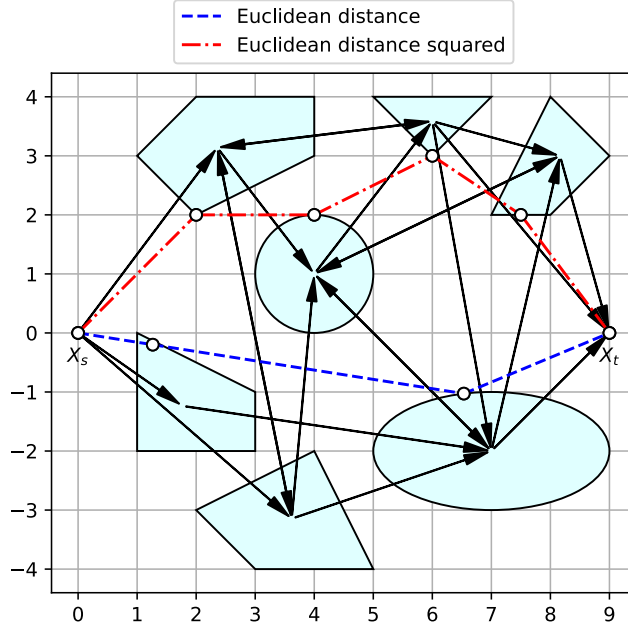


Figure 5: Graph G and sets X_v for the example in Section 11.1. The blue dashed (red dash-dotted) line is the shortest path in case the edge length is the Euclidean distance (2) (Euclidean distance squared (3)).

11 Numerical Results

In this section we collect multiple numerical examples. We start in Section 11.1 with a two-dimensional problem. Section 11.2 contains a statistical analysis of the performance of our formulation in case of large-scale random instances of the SPP (1). In Section 11.3 we demonstrate the applicability of our problem formulation to optimal control of PWA systems. We conclude in Section 11.4 presenting two SPP instances that are adversarially designed to exhibit two weaknesses of our MICP.

The code necessary to reproduce all the results presented in this section can be found at [59]. It uses **Drake** [90] as an interface to the commercial solver **Mosek** 9.2.33 [69]. The solution times we report are retrieved through **Mosek**'s attribute `MSK_DINF_OPTIMIZER_TIME`, and they are obtained with default options on a machine with processor 2.4 GHz 8-Core Intel Core i9 and memory 64 GB 2667 MHz DDR4.

11.1 Two-Dimensional Example

The first example we consider is the two-dimensional SPP depicted in Figure 5. We have a graph G with $|V| = 9$ vertices, $|E| = 22$ edges, and multiple cycles. The source $X_s = \{\theta_s\}$ and target $X_t = \{\theta_t\}$ sets are single points, while the remaining regions are full dimensional. The geometry of the sets X_v and the edge set E can be deduced from Figure 5. We consider two edge-length functions: the Euclidean distance (2) and the Euclidean distance squared (3). In both cases the resulting optimization problem is a Mixed-Integer Second-Order-Cone Program (MISOCP). The corresponding shortest paths are shown in Figure 5 as a blue dashed line and a red dash-dotted line. As expected, the first path is almost straight while the length of the segments in the second is better balanced.

In Section 3 we have seen that the size of the sets X_v can have dramatic effects on the hardness of the SPP (1): the SPP is easily solvable when the sets collapse to singletons, while the combination of large sets X_v , cycles, and non-homogeneous edge lengths can make the SPP very hard. Motivated

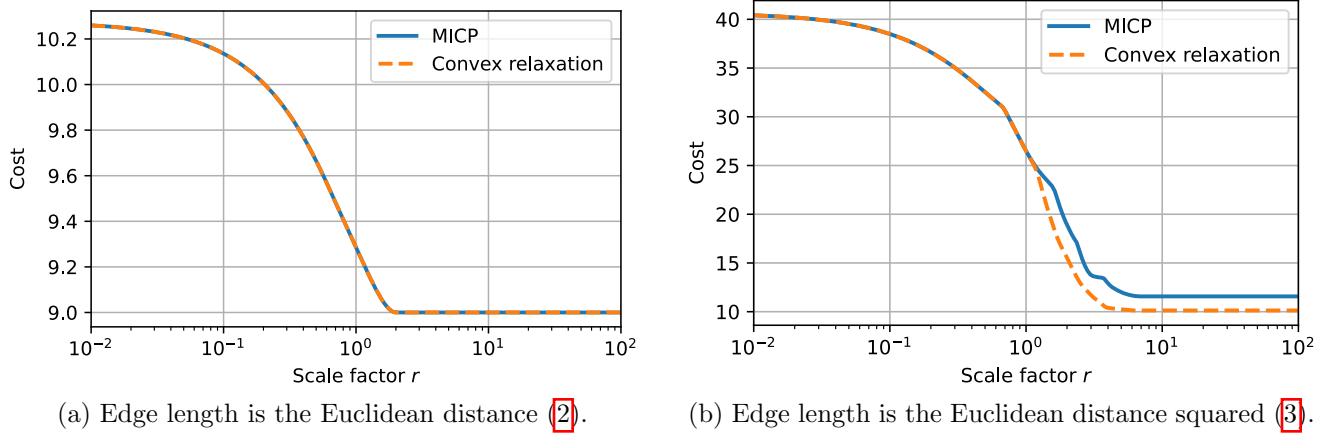


Figure 6: Numerical results from Section 11.1. Optimal cost of the MICP (18) and of its convex relaxation as functions of the size r of the sets X_v .

by this observation, in Figure 6 we compare the cost of the MICP (21) and of its convex relaxation as functions of the size of the regions X_v . We control the size of these sets via the scalar $r > 0$. The nominal case ($r = 1$) is depicted in Figure 5. For $r \neq 1$ each set X_v is shrunk or enlarged via a uniform scaling, with scale factor r , that fixes the Chebyshev center of the set (center of the largest inscribed ball).

When the edge length is the Euclidean distance (2), Figure 6a shows that the convex relaxation is tight for all values of r . This agrees with the results from Section 6.2.1 according to which our MICP is tight when the sets are small. On the other hand, that the relaxation gap is zero even for nonzero r is not an obvious result.

In light of what we said and noticing that the graph in Figure 5 is not Hamiltonian, we do not expect our formulation to perform equally well in case of the non-homogeneous edge length (3). In this case, Figure 6b shows that the convex relaxation is not always tight, even though the relaxation gap is small even in the worst case. For small scales r , the results from Section 6.2.1 imply once again that the relaxation gap must be zero. For larger r , approximately $r > 1$, the convex relaxation becomes slightly loose. Equation (39) can be used to predict the asymptotic cost of the MICP: $\|\theta_t - \theta_s\|_2^2 / K = 11.6$, where $K = 7$ is the maximum number of edges traversed by an s - t path in the graph from Figure 5. A closer inspection of Figure 6b reveals that the curve of the convex relaxation converges to $\|\theta_t - \theta_s\|_2^2 / (|V| - 1) = 10.1$, which corresponds to the lower bound derived in Proposition 5.

In many circumstances the convex relaxation of our MICP yields tighter lower bounds than the one from Proposition 5. As an example, the removal of the edge connecting the top-left set to the bottom set in Figure 5 does not make the graph Hamiltonian, but it is sufficient to close the asymptotic relaxation gap in Figure 6b.

11.2 Large-Scale Random Instances

In the previous example we have tested our MICP on a small-scale SPP and we have analyzed its strength only as a function of the size of the sets X_v . We now move to problems of larger scale, and we analyze the impact of multiple parameters on the efficiency of our formulation. We generate a large number of random problem instances and we illustrate the resulting solution statistics.

Remark 17. Generating random graphs representative of the “typical” SPP on convex sets we might

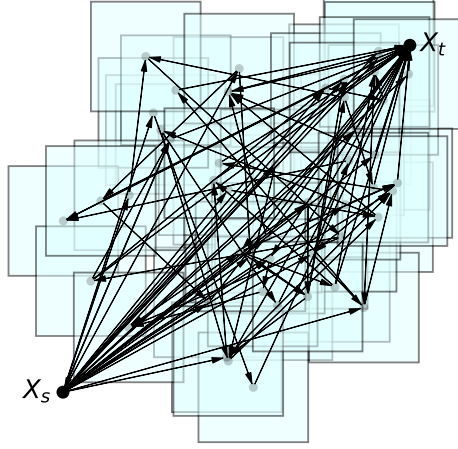


Figure 7: Projection onto two dimensions of a random instance of the SPP from Section 11.2 for a nominal value of the problem parameters.

encounter in practice is a difficult operation. Restrictions such as requiring the source s to be connected to all the vertices in the graph introduce strong biases in the topology of the graph. Inevitably, the instances we describe below are not completely representative, and our algorithm might perform worse or better on different classes of random graphs. With the following results we do not want to make any claim regarding, e.g., the average strength of our formulation. Our purpose is instead to show that the applicability of our formulation is not limited to small-scale problems.

We construct a random instance of problem (1) as follows. We set $X_s := \{0 \in \mathbb{R}^d\}$ and $X_t := \{1 \in \mathbb{R}^d\}$. Each of the remaining $|V| - 2$ sets X_v is an axis-aligned cube of volume Λ with center drawn from the uniform distribution over $[0, 1]^d$. Given a number of edges $|E|$, we construct the edge set E in two steps. First we generate multiple s - t paths such that each vertex in $V - \{s, t\}$ is traversed exactly by one path. These are determined via a random partition of the set $V - \{s, t\}$: the number of sets in the partition (number of paths) is drawn uniformly from the interval $[1, |V| - 2]$, and also the number of vertices in each set (length of each path) is a uniform random variable. Secondly, we expand the edge set E by drawing edges uniformly at random from the set $\{(u, v) \in V^2 : u \neq v\}$ until the desired cardinality $|E|$ is reached. We use the following nominal parameters: $d = 4$ dimensions, $|E| = 100$ edges, $|V| = 50$ regions, and a volume $\Lambda = 0.01$ for the regions X_v . To give an idea of what these problems look like, the projection onto two dimensions of a random instance generated with these parameters is shown in Figure 7.

As edge lengths, we consider the Euclidean distance (2) and the Euclidean distance squared (3). For each edge length, first we solve 100 random instances with nominal parameters. Then we consider four subgroups of the parameters: for each subgroup, we multiply the value of the parameters in it by 5, and we solve another 100 random instances. Table 1 shows the statistics of these trials: the two groups of columns report the median and maximum relaxation gap and MICP solution time. In support of the analysis below, we recall that both the edge lengths (2) and (3) lead to an MISOCP, and that these problems have $|E|$ binaries, $O(d|E|)$ continuous variables, and $O(d(|V| + |E|))$ constraints.

Overall, the Euclidean edge length (2) results in easier optimizations: the relaxation gaps never exceed 1.24% and computation times are relatively low.

The squared edge length (3) leads to more challenging problems even though, in the nominal case, the average relaxation gap is still very low and the computation times are always within 0.6 s. The growth of the space dimension to $d = 20$ increases the size of our programs, and also deteriorates

Increased parameters	Relaxation gap (%)				MICP solve time (s)			
	Eucl.		Eucl. sq.		Eucl.		Eucl. sq.	
	med	max	med	max	med	max	med	max
None (nominal)	0.00	0.34	0.0	2.1	0.08	0.43	0.1	0.6
Dimensions d	0.00	0.21	7.4	28.9	0.55	5.26	2.3	133.3
Edges $ E $	0.00	1.24	14.0	32.9	1.65	12.74	22.5	197.2
Vertices $ V $, edges $ E $	0.00	0.25	0.00	5.3	1.14	4.32	1.1	5.3
Volume Λ	0.00	0.56	0.0	9.1	0.13	0.68	0.1	0.9

Table 1: Relaxation gap and computation times, in the median and worst case, for the random problem instances described in Section 11.2. First row: solution statistics for 100 problem instances with nominal parameters ($d = 4$, $|E| = 100$, $|V| = 50$, and $\Lambda = 0.01$). Remaining rows: solution statistics for 100 problem instances with a subset of the parameters increased by a factor of 5. Two edge-length functions are considered: the Euclidean distance (2) and the Euclidean distance squared (3). These results show that our formulation can tackle problems of significant size; however, given the random nature of these instances, these values might be unrepresentative of the average performance of our MICP.

the tightness of the convex relaxation. In the worst case, we have a relaxation gap of 28.9% and a solution time greater than 2 min. A similar analysis applies when the number $|E|$ of edges is increased from 100 to 500: in general, we have found our formulation to struggle with graphs of high density of edges $|E|/|V|$. To show this, in the fourth row we keep $|E| = 500$ edges but we increase the vertices to $|V| = 250$: this has the effect of reducing the edge density and, even if the resulting MICPs are bigger than the ones from the previous case, the relaxation gap and the computation times are strongly reduced. Finally, we increase the volume of the cubes X_v from $\Lambda = 0.01$ to $\Lambda = 0.05$: these sets have now a total volume of $|V|\Lambda = 2.5$, which is significantly larger than the unit cube containing them. Despite this, the performance of our formulation does not differ significantly from the nominal case. Note that this does not contradict the previous example, where we were analyzing the regime of extremely large sets X_v ; recall also that the volume of the sets does not affect the size of the MICPs.

Finally, we report that the solution times for the convex (second-order-cone) relaxations of these programs are of the order of hundredths of second or, at most, a few tenths of second.

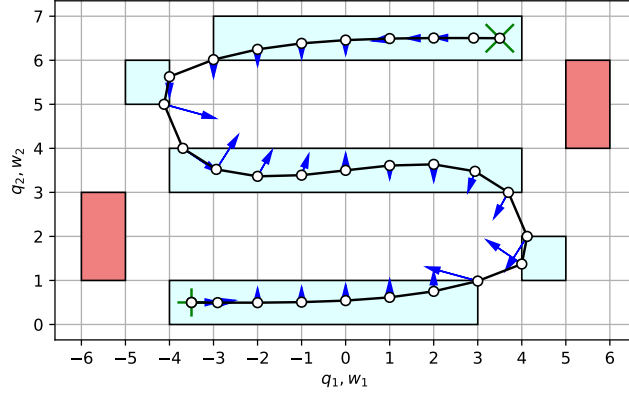
11.3 Optimal Control of a Piecewise-Affine System

This example illustrates the discussion from Section 9 via the optimal-control problem shown in Figure 8a. We consider a mechanical system with position $q \in \mathbb{R}^2$ and velocity $\dot{q} \in \mathbb{R}^2$. The force $w \in \mathbb{R}^2$ serves as control input. The system has the dynamics of a double integrator

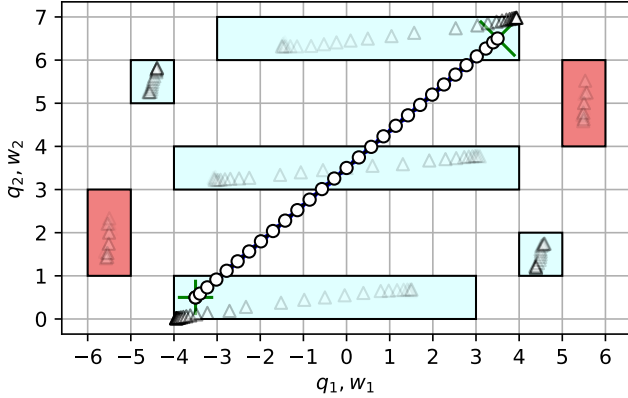
$$q(k+1) = q(k) + \dot{q}(k), \quad \dot{q}(k+1) = \dot{q}(k) + \eta w(k), \quad (47)$$

where η is a scalar parameter that regulates the controllability of the system. We represent the state of the system as $\zeta := (q, \dot{q})$.

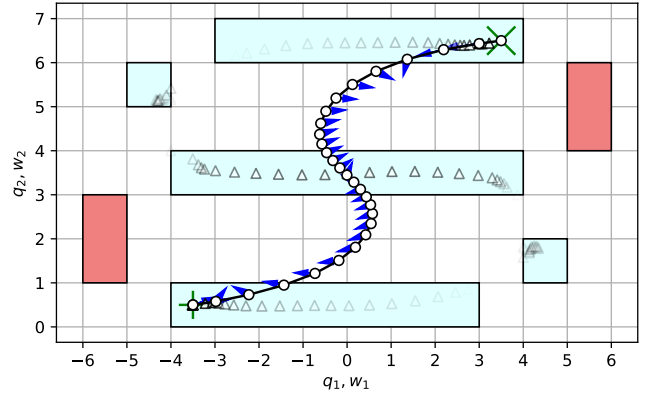
At time $k = 1$, the system is in position $q(1) := (-3.5, 0.5)$ (bottom-left green plus in Figure 8) with velocity $\dot{q}(1) := (0, 0)$. At each time step $k = 1, \dots, K-1$, the position vector $q(k)$ is allowed to be in one of the seven regions depicted in Figure 8, while the velocity and the controls are limited by the constraints $\|\dot{q}(k)\|_\infty \leq 1$ and $\|w(k)\|_\infty \leq 1$. The goal is to reach the configuration $q(K) := (3.5, 6.5)$ (top-right green cross in Figure 8) with zero velocity $\dot{q}(K)$ in $K := 30$ time steps. In doing this, we



(a) Optimal solution of the MICP.



(b) Solution of the convex relaxation of the formulation [68, 61]. The relaxation gap is 93%, and the corresponding MICP is solved in 218 s.



(c) Solution of the convex relaxation of the proposed formulation. The relaxation gap is 20%, and the corresponding MICP is solved in 1.3 s.

Figure 8: Control problem of driving a second-order system from start (green plus) to goal (green cross). In the light-blue regions the system is highly controllable ($\eta = 1$) and in the red regions controllability is low ($\eta = 0.1$). Optimal positions $(q(k))_{k=1}^K$ are white circles, optimal controls $(w(k))_{k=1}^{K-1}$ are blue arrows. The triangles represent the auxiliary variables $q_i(k)$ whose convex combination yields $q(k)$. The opacity of the triangles equals the optimal value of the indicator variables $b_i(k)$ which serve as weights in this convex combination.

want minimize the quadratic form

$$\sum_{k=0}^{K-1} \left(\frac{1}{5} \|\dot{q}(k)\|_2^2 + \|w(k)\|_2^2 \right). \quad (48)$$

We let the controllability parameter η vary between the regions. For the regions included in the range $-5 \leq q_1 \leq 5$ (light blue in Figure 8) we set $\eta = 1$, and the system is highly controllable. In the two remaining two regions (red in Figure 8) we let $\eta = 0.1$, making it very expensive to apply any significant force. Since the parameter η varies with the system state, the dynamics in (47) are PWA and the control problem falls into the class considered in Section 9.2. The graph G beneath this problem (see Figure 4b) has $|V| = 205$ vertices and $|E| = 1386$ edges, the convex sets X_v live in a space of $d = 6$ dimensions. Because of the quadratic objective (48), the resulting optimization problem is an MISOCP.

Figure 8a shows the optimal trajectory $(q(k))_{k=1}^K$ (white circles) and the optimal controls $(w(k))_{k=1}^{K-1}$ (blue arrows). Geometrically, the red regions would be shortcuts towards the goal, but the low controllability in these areas makes it very expensive not to fall out of the feasible set. The optimal strategy is then to follow the S-shaped path and incur a cost of 9.74.

As a benchmark for our SPP formulation, we first solve this problem using the strongest formulation available in the control literature: this also employs perspective functions and has been presented in [68] and further analyzed in [61] Section 5.2.2]. For each time step k , it uses $|I|$ indicator variables $b_i(k) \in \{0, 1\}$ to select which one of the affine dynamics in (41) should be applied. This is done by decomposing the state and the controls at time k into the convex combination of $|I|$ auxiliary variables $(\zeta_i(k), w_i(k)) \in D_i$:

$$(\zeta(k), u(k)) = \sum_{i \in I} b_i(k) (\zeta_i(k), w_i(k)), \quad (49)$$

where $\sum_{i \in I} b_i(k) = 1$.¹⁰ Each copy $(\zeta_i(k), w_i(k))$ is used to predict the next state according to the i th dynamics, and the actual state of the system at time $k + 1$ is recovered as the convex combination of these predictions:

$$\zeta(k + 1) = \sum_{i \in I} b_i(k) (A_i \zeta_i(k) + B_i w_i(k) + c_i). \quad (50)$$

When the binaries are relaxed, $b_i(k) \in [0, 1]$, the system evolves according to a convex combination of the various affine dynamics. Also this formulation yields an MISOCP (this is due to a perspective reformulation of the objective function [68, 61]).

Figure 8b shows the solution of the convex relaxation of the formulation [68, 61]. It reports the position $q(k)$, the (barely visible) controls $w(k)$, and the auxiliary copies $q_i(k)$ of the position vector whose convex combination yields $q(k)$. The latter have triangle markers with opacity equal to the value of the indicator $b_i(k)$. At each time step k , the solver is allowed to select the best convex combination of the $|I|$ affine dynamics: it decides to reach the goal with a perfectly-straight trajectory and incur a cost of 0.70, which is only 7% of the MICP cost (93% relaxation gap). Note that this behavior is completely insensitive to the geometry of the problem. The auxiliary variables are also uninformative: given the wide variety of convex combinations of the affine dynamics that yield a straight trajectory, the values of $q_i(k)$ and $b_i(k)$ make it impossible to guess in which region the system should be at a given time. The solver does not even realize that stepping in the regions of low controllability is suboptimal, and it assigns nonzero weights $b_i(k)$ to these regions (visible triangle markers in the red regions). The MICP resulting from this problem formulation requires 218 s to be solved to global optimality.

The convex relaxation of our formulation is much tighter: its optimal value is 7.84, which is 80% of the MICP cost (20% relaxation gap). This has a dramatic effect on computation times which are now reduced to 1.3 s.

To make a plot comparable with 8b we leverage the structure of the graph G in Figure 4b. For each time step $k = 1, \dots, K - 1$, this graph has vertices $v = (k, i)$ and the total flow

$$b_i(k) := \sum_{e \in E_v^{\text{out}}} \varphi_e \quad (51)$$

traversing vertex v takes the role of the binary indicator from the formulation [68, 61].¹¹ In fact, at optimality of the MICP (21) we have $(\zeta(k), w(k)) \in D_i$ if and only if $b_i(k) = 1$, while the conservation of

¹⁰The bilinear constraints (49) and (50) are convexified using the method from [3, 13].

¹¹By the conservation of flow in (21c) and its spatial version (21d), we could equivalently define $b_i(k) := \sum_{e \in E_v^{\text{in}}} \varphi_e$ and $(\zeta_i(k), w_i(k)) := \sum_{e \in E_v^{\text{in}}} z_e / b_i(k)$ in (51) and (52).

flow in (21c) is easily seen to imply $\sum_{i \in I} b_i(k) = 1$. Recalling that the vertex positions x_v are obtained by stacking the state and the controls, and provided that $b_i(k) > 0$, the role of the auxiliary continuous variables from [68][61] can be taken by

$$(\zeta_i(k), w_i(k)) := \frac{1}{b_i(k)} \sum_{e \in E_v^{\text{out}}} y_e \in D_i, \quad (52)$$

where the inclusion on the right holds even for nonbinary flows and follows directly from the nonnegativity constraint (21b). Reconstructing the system state and controls as in (49), we obtain

$$(\zeta(k), w(k)) = \sum_{v \in \{k\} \times I} \sum_{e \in E_v^{\text{out}}} y_e.$$

The values just described are depicted in Figure 8c. The system trajectory $(q(k))_{k=1}^K$ and the controls $(w(k))_{k=1}^{K-1}$ obtained from the proposed convex program resemble the S-shaped optimal solution in Figure 8a much more closely than the formulation [68][61]. For the auxiliary variables, we note that all the markers in the regions with low controllability are invisible, meaning that our convex relaxation identifies these as regions of high cost, and sets to zero the corresponding indicators $b_i(k)$. All the visible points $q_i(k)$ are clustered along the optimal trajectory of the MICP, suggesting that our convex relaxation contains detailed information on the optimal path to reach the goal.

11.4 Adversarial Instances

We conclude with two examples that illustrate carefully-chosen scenarios in which the convex relaxation of the MICP (21) is not tight. This first has to do with symmetries in the graph G , the second with cycles.

11.4.1 Symmetries in the Graph

We consider the SPP problem depicted in Figure 9a. We have a graph G with $|V| = 5$ vertices and $|E| = 5$ edges. All the sets X_v are singletons $\{\theta_v\}$, except for X_3 which is a full-dimensional rectangle (light blue in Figure 9a). The edge lengths penalize the Euclidean distance between the vertices as in (2). Solving this SPP, we obtain the optimal path $\pi = (s, 1, 3, t)$ with length 7.39. The corresponding vertex positions are connected by a blue dashed line in Figure 9a. Notice that, because of the problem symmetry, the solution $\pi = (s, 2, 3, t)$ is also optimal.

Figure 9b illustrates the solution of the convex relaxation of the MICP. For each edge $e \in E$, we connect the optimal location of the points $\bar{y}_e := y_e/\varphi_e$ and $\bar{z}_e := z_e/\varphi_e$ with a blue dashed line, labelled in red with the corresponding flow φ_e . Note that, for $\varphi_e > 0$, the points $\bar{y}_e$ and $\bar{z}_e$ represent the actual values of x_u and x_v based on which the cost of the edge $e = (u, v)$ is computed; in fact, we have $\tilde{\ell}_e(y_e, z_e, \varphi_e) = \ell_e(\bar{y}_e, \bar{z}_e)\varphi_e$. As opposed to the MICP, the convex relaxation splits the unit of flow injected in the source s into two: half unit is shipped to the target t via the upper path, the other half via the bottom path. The optimal value of this convex program is 7.00.

The looseness of the convex relaxation can be explained as follows. If we denote with α the flow traversing edge $(1, 3)$, the conservation of flow in (21c) implies $\varphi_{(2,3)} = 1 - \alpha$. Similarly, the flow through the edge $(3, t)$ is always one. For a positive flow, the nonnegativity constraints (21b) give $\bar{y}_e \in X_u$ and $\bar{z}_e \in X_v$, and they force the variables $\bar{y}_{(1,3)}$, $\bar{y}_{(2,3)}$, and $\bar{z}_{(3,t)}$ to match $\theta_1 = (0, 2)$, $\theta_2 = (0, -2)$, and $\theta_t = (5, 0)$, respectively. The cost terms in (21a) corresponding to the edges $(1, 3)$, $(2, 3)$, and $(3, t)$ are

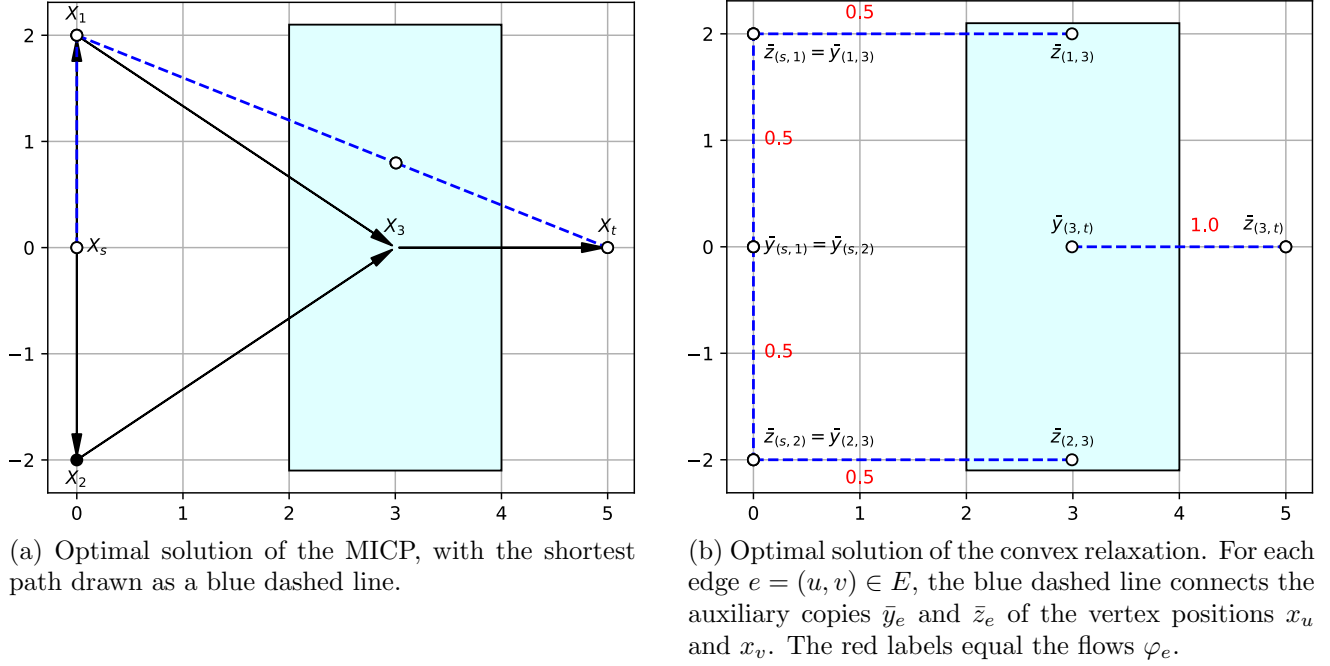


Figure 9: Example from Section 11.4.1 that describes how symmetries in the graph G can loosen the convex relaxation of our MICP. For the convex relaxation, the cost contribution of edge e is obtained by multiplying the flow φ_e by the distance between $\bar{y}_e$ and $\bar{z}_e$. Because of the symmetry of this SPP, the constraints in the convex program only require the mean of $\bar{z}_{(1,3)}$ and $\bar{z}_{(2,3)}$ to match $\bar{y}_{(3,t)}$. The cost is then reduced by separating the first two points vertically.

then

$$\|\bar{z}_{(1,3)} - \theta_1\|_2 \alpha + \|\bar{z}_{(2,3)} - \theta_2\|_2 (1 - \alpha) + \|\theta_t - \bar{y}_{(3,t)}\|_2. \quad (53)$$

The only constraint that links these variables is the spatial conservation of flow (21d) at vertex 3, which reads

$$\alpha \bar{z}_{(1,3)} + (1 - \alpha) \bar{z}_{(2,3)} = \bar{y}_{(3,t)}.$$

When α is set to 0.5, this constraint only requires the mean of $\bar{z}_{(1,3)}$ and $\bar{z}_{(2,3)}$ to match $\bar{y}_{(3,t)}$, instead of forcing one of the first two points to match the third, as it would be for $\alpha \in \{0, 1\}$. Therefore, while keeping their mean equal to $\bar{y}_{(3,t)}$, the points $\bar{z}_{(1,3)}$ and $\bar{z}_{(2,3)}$ can move vertically to get closer to θ_1 and θ_2 , respectively. This minimizes the first two terms in (53), and keeps the third unchanged. (Note that, if these points were to move horizontally, the variations of the three terms in (53) would cancel, and the cost would not change.)

To sum up, when in an SPP (1) multiple near-optimal paths merge in a single large region X_v , the auxiliary copies of x_v can scatter to decrease the cost terms in (21a), while verifying the spatial conservation of flow (21d). Note also that, even though the relaxation gap for the example we just analyzed is small (5.2%), a careful redesign of the edge lengths ℓ_e and the position of the sets X_v can make it arbitrarily large.

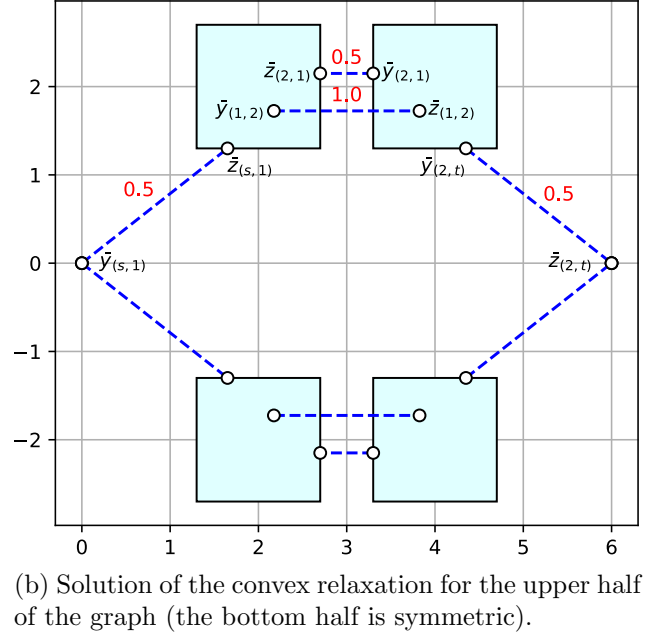
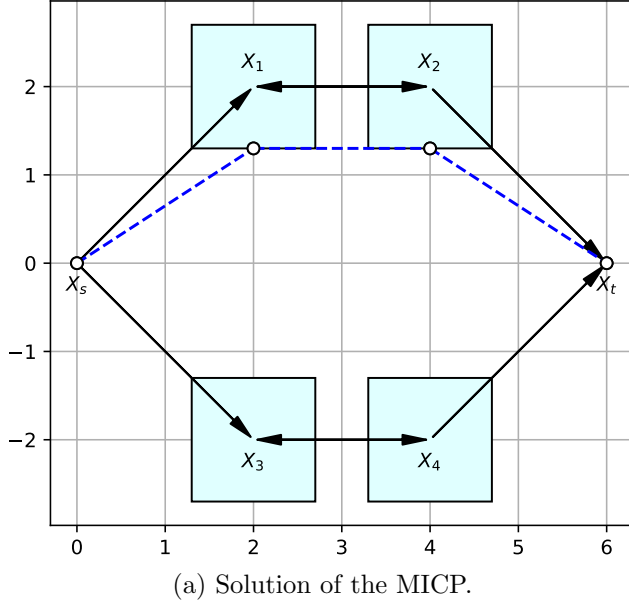


Figure 10: Example from Section 11.4.2 that describes how cycles and nonhomogeneous edge lengths can make the convex relaxation of our MICP loose. By pushing a nonzero flow in the edge $(2, 1)$, the convex relaxation divides the three steps from the solution of the MICP in a larger number of smaller steps. Since the edge length (3) minimizes the squares of the step lengths, this turns out to be advantageous.

11.4.2 Cycles in the Graph

The second case we discuss ties back to the complexity analysis from Section 3 and it shows how cycles and nonhomogeneous edge lengths can make the convex relaxation of our MICP loose. We analyze the SPP in Figure 10a: we have a graph G with $|V| = 6$ vertices, $|E| = 8$ edges, and two cycles $\{(1, 2), (2, 1)\}$ and $\{(3, 4), (4, 3)\}$. The source and target sets are singletons, $X_s := \{\theta_s\}$ and $X_t := \{\theta_t\}$; the remaining sets are full dimensional. The edge length is the Euclidean distance squared (3). The path $\pi = (s, 1, 2, t)$ is one of the two optimal solutions of this SPP: it has length 15.38 and it is depicted in Figure 10a. The solution of the convex relaxation is illustrated in Figure 10b and has cost 14.64 (only the upper half of the solution is reported, the bottom half is symmetric).

The convex relaxation asks to ship a flow of 0.5 along the edge $(2, 1)$, even though a nonzero flow $\varphi_{(2,1)}$ is clearly suboptimal for the MICP. This behavior is motivated as follows. If the edge $(2, 1)$ had zero flow, the spatial conservation of flow (21d) would imply $\bar{z}_{(s,1)} = \bar{y}_{(1,2)}$ and $\bar{z}_{(1,2)} = \bar{y}_{(2,t)}$, and the convex relaxation would be tight. On the other hand, similarly to the previous example, by pushing a flow of 0.5 along the edge $(2, 1)$, the spatial conservation of flow forces only the mean of the auxiliary copies of x_1 and x_2 to coincide:

$$(\bar{z}_{(s,1)} + \bar{z}_{(2,1)})/2 = \bar{y}_{(1,2)}, \quad \bar{z}_{(1,2)} = (\bar{y}_{(2,1)} + \bar{y}_{(2,t)})/2.$$

By separating the variables $\bar{z}_{(s,1)}$ and $\bar{z}_{(2,1)}$, as well as $\bar{y}_{(2,1)}$ and $\bar{y}_{(2,t)}$, the three line segments in the solution of the MICP from Figure 10a can be divided in a larger number of smaller segments. Since the objective is to minimize the sum of the squared lengths of these segments, this turns out to be advantageous. More precisely, the optimal value of the MICP is

$$\|x_1 - \theta_s\|_2^2 + \|x_2 - x_1\|_2^2 + \|\theta_t - x_2\|_2^2 = 5.69 + 4.00 + 5.69 = 15.38, \quad (54)$$

while, using the symmetry of the solution, the cost of the convex relaxation can be verified to be

$$\begin{aligned} & 2(0.5\|\bar{z}_{(s,1)} - \theta_s\|_2^2 + \|\bar{z}_{(1,2)} - \bar{y}_{(1,2)}\|_2^2 + 0.5\|\bar{z}_{(2,1)} - \bar{y}_{(2,1)}\|_2^2 + 0.5\|\theta_t - \bar{y}_{(2,t)}\|_2^2) \\ & = 2(2.21 + 2.72 + 0.18 + 2.21) = 14.64. \end{aligned} \quad (55)$$

The terms in (55) are greater in number but smaller in magnitude than the ones in (54), leading overall to a nonzero relaxation gap. As in the previous example, this gap can easily be increased, e.g. by enlarging the full-dimensional sets X_v .

To sum up: in case of nonhomogeneous edge lengths, the convex relaxation of our MICP might find it convenient to push a nonzero flow along a cycle and fragment the optimal MICP path into a larger number of smaller segments.

Remark 18. Figure 10b shows that the degree constraints in (21c) actively limit to one the total flow traversing the vertices 1 and 2 (as well as 3 and 4). If we were to remove these constraints, the cost of the convex relaxation would drop; showing once again that, even if redundant for the original LP (5), these constraints strengthen our MICP. The degree constraints are also the reason why, in this example, we need two s - t paths for the analyzed behavior to emerge: if the graph was limited to the upper half, we would have only one feasible flow ($\varphi_{(s,1)} = \varphi_{(1,2)} = \varphi_{(2,t)} = 1$ and $\varphi_{(2,1)} = 0$) and the convex relaxation would be tight.

12 Conclusions and Future Works

We have analyzed a generalization of the SPP in which the position of each vertex in the graph is a continuous decision variable lying in a convex set, and the length of an edge is a convex function of the position of the vertices it connects. Our main contribution is a strong mixed-integer convex formulation for the solution of this NP-hard problem. A wide variety of numerical tests show that the convex relaxation of this MICP is often very tight. We have focused part of our attention on control systems: many mixed-integer control problems turn out to be interpretable as SPPs and, in our tests, the proposed MICP outperforms state-of-the-art techniques for their solution.

Currently, we are working on benchmarking the techniques proposed here for the SPP against existing mathematical-programming formulations of other graph problems with neighborhoods. In the future, we plan to work on the development of approximation algorithms for the SPP in graphs of convex sets. To this end, the fact that the proposed MICP shares the same structure as the LP formulation of the classical SPP might allow us to leverage a massive body of works (see, e.g., [95]). On the other hand, negative results on the hardness of approximating the longest path in a directed graph [5], a problem which we have seen being a special case of our SPP, suggest that further assumptions on the structure of our problem will be needed to progress in this direction. In terms of applications, we are currently developing high-performance algorithms for robot motion planning that are based on the results presented in this paper.

Acknowledgements

The authors would like to thank Hongkai Dai for all the time spent improving the optimization-solver interface employed in the numerical examples presented in this paper.

This research was supported by the National Science Foundation, Award No. EFMA-1830901, and by the Department of the Navy, Office of Naval Research, Award No. N00014-18-1-2210. Any opinions,

findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the Office of Naval Research.

References

- [1] Ravindra K. Ahuja, Thomas L. Magnanti, and James B. Orlin. *Network Flows: theory, algorithms, and applications*. Prentice-Hall, 1993.
- [2] Esther M Arkin and Refael Hassin. Approximation algorithms for the geometric covering salesman problem. *Discrete Applied Mathematics*, 55(3):197–218, 1994.
- [3] Egon Balas. Disjunctive programming: Properties of the convex hull of feasible points. *Discrete Applied Mathematics*, 89(1-3):3–44, 1998.
- [4] Alberto Bemporad and Manfred Morari. Control of systems integrating logic, dynamics, and constraints. *Automatica*, 35(3):407–427, 1999.
- [5] Andreas Björklund, Thore Husfeldt, and Sanjeev Khanna. Approximating longest directed paths and cycles. In *International Colloquium on Automata, Languages, and Programming*, pages 222–233. Springer, 2004.
- [6] Víctor Blanco, Elena Fernández, and Justo Puerto. Minimum spanning trees with neighborhoods: Mathematical programming formulations and solution methods. *European Journal of Operational Research*, 262(3):863–878, 2017.
- [7] Richard Bormann, Florian Jordan, Joshua Hampp, and Martin Hägele. Indoor coverage path planning: Survey, implementation, analysis. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1718–1725. IEEE, 2018.
- [8] Francesco Borrelli, Alberto Bemporad, Michael Fodor, and Davor Hrovat. An MPC/hybrid system approach to traction control. *IEEE Transactions on Control Systems Technology*, 14(3):541–552, 2006.
- [9] Stephen Boyd and Lieven Vandenbergh. *Convex optimization*. Cambridge university press, 2004.
- [10] Joel W Burdick, Amanda Bouman, and Elon Rimon. From multi-target sensory coverage to complete sensory coverage: An optimization-based robotic sensory coverage approach. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 10994–11000. IEEE, 2021.
- [11] Eduardo F Camacho, Daniel R Ramírez, Daniel Limón, D Muñoz De La Peña, and Teodoro Alamo. Model predictive control techniques for hybrid systems. *Annual reviews in control*, 34(1):21–31, 2010.
- [12] John Canny and John Reif. New lower bound techniques for robot motion planning problems. In *28th Annual Symposium on Foundations of Computer Science (sfcs 1987)*, pages 49–60. IEEE, 1987.
- [13] Sebastián Ceria and João Soares. Convex programming for disjunctive convex optimization. *Mathematical Programming*, 86(3):595–614, 1999.

- [14] Wei-pang Chin and Simeon Ntafos. Optimum watchman routes. In *Proceedings of the second annual symposium on Computational geometry*, pages 24–33, 1986.
- [15] Howie Choset. Coverage for robotics—a survey of recent results. *Annals of mathematics and artificial intelligence*, 31(1):113–126, 2001.
- [16] Patrick L Combettes. Perspective functions: Properties, constructions, and examples. *Set-Valued and Variational Analysis*, 26(2):247–264, 2018.
- [17] Michele Conforti, Gérard Cornuéjols, and Giacomo Zambelli. Extended formulations in combinatorial optimization. *4OR*, 8(1):1–48, 2010.
- [18] George Dantzig, Ray Fulkerson, and Selmer Johnson. Solution of a large-scale traveling-salesman problem. *Journal of the operations research society of America*, 2(4):393–410, 1954.
- [19] Robin Deits and Russ Tedrake. Footstep planning on uneven terrain with mixed-integer convex optimization. In *2014 IEEE-RAS international conference on humanoid robots*, pages 279–286. IEEE, 2014.
- [20] Robin Deits and Russ Tedrake. Computing large convex regions of obstacle-free space through semidefinite programming. In *Algorithmic foundations of robotics XI*, pages 109–124. Springer, 2015.
- [21] Marc Demange, Tınaz Ekim, Bernard Ries, and Cerasela Tanasescu. On some applications of the selective graph coloring problem. *European Journal of Operational Research*, 240(2):307–314, 2015.
- [22] Marc Demange, Jérôme Monnot, Petrica Pop, and Bernard Ries. On the complexity of the selective graph coloring problem in some special classes of graphs. *Theoretical Computer Science*, 540:89–102, 2014.
- [23] Ashwin Deshpande. *Exact geometry algorithms for robotic motion planning*. PhD thesis, Massachusetts Institute of Technology, 2019.
- [24] Edsger W Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1(1):269–271, 1959.
- [25] Yann Disser, Matúš Mihalák, Sandro Montanari, and Peter Widmayer. Rectilinear shortest path and rectilinear minimum spanning tree with neighborhoods. In *International Symposium on Combinatorial Optimization*, pages 208–220. Springer, 2014.
- [26] Moshe Dror, Alon Efrat, Anna Lubiw, and Joseph SB Mitchell. Touring a sequence of polygons. In *Proceedings of the thirty-fifth annual ACM symposium on Theory of computing*, pages 473–482, 2003.
- [27] Moshe Dror and Mohamed Haouari. Generalized Steiner problems and other variants. *Journal of Combinatorial Optimization*, 4(4):415–436, 2000.
- [28] Moshe Dror, Mohamed Haouari, and J Chaouachi. Generalized spanning trees. *European Journal of Operational Research*, 120(3):583–592, 2000.
- [29] Matthew G Earl and Raffaello D’Andrea. Modeling and control of a multi-agent system using mixed integer linear programming. In *Proceedings of the 41st IEEE Conference on Decision and Control, 2002.*, volume 1, pages 107–111. IEEE, 2002.

- [30] Corinne Feremans, Martine Labbé, and Gilbert Laporte. Generalized network design problems. *European Journal of Operational Research*, 148(1):1–13, 2003.
- [31] Corinne Feremans, Martine Labbé, and Gilbert Laporte. The generalized minimum spanning tree problem: Polyhedral analysis and branch-and-cut algorithm. *Networks: An International Journal*, 43(2):71–86, 2004.
- [32] Matteo Fischetti, Juan José Salazar González, and Paolo Toth. The symmetric generalized traveling salesman polytope. *Networks*, 26(2):113–123, 1995.
- [33] Matteo Fischetti, Juan José Salazar González, and Paolo Toth. A branch-and-cut algorithm for the symmetric generalized traveling salesman problem. *Operations Research*, 45(3):378–394, 1997.
- [34] Antonio Frangioni and Claudio Gentile. Perspective cuts for a class of convex 0–1 mixed integer programs. *Mathematical Programming*, 106(2):225–236, 2006.
- [35] Enric Galceran and Marc Carreras. A survey on coverage path planning for robotics. *Robotics and Autonomous systems*, 61(12):1258–1276, 2013.
- [36] Iacopo Gentilini, François Margot, and Kenji Shimada. The travelling salesman problem with neighbourhoods: MINLP solution. *Optimization Methods and Software*, 28(2):364–378, 2013.
- [37] Tobias Geyer, Georgios Papafotiou, and Manfred Morari. Hybrid model predictive control of the step-down DC–DC converter. *IEEE Transactions on Control Systems Technology*, 16(6):1112–1124, 2008.
- [38] Gianpaolo Ghiani and Gennaro Improta. An efficient transformation of the generalized vehicle routing problem. *European Journal of Operational Research*, 122(1):11–17, 2000.
- [39] Oktay Günlük and Jeff Linderoth. Perspective reformulations of mixed integer nonlinear programs with indicator variables. *Mathematical programming*, 124(1-2):183–205, 2010.
- [40] Oktay Günlük and Jeff Linderoth. Perspective reformulation and applications. In *Mixed Integer Nonlinear Programming*, pages 61–89. Springer, 2012.
- [41] Weiqiao Han and Russ Tedrake. Feedback design for multi-contact push recovery via LMI approximation of the piecewise-affine quadratic regulator. In *2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)*, pages 842–849. IEEE, 2017.
- [42] Wilhemus PMH Heemels, Bart De Schutter, and Alberto Bemporad. Equivalence of hybrid dynamical models. *Automatica*, 37(7):1085–1091, 2001.
- [43] Pedro Hespanhol, Rien Quirynen, and Stefano Di Cairano. A structure exploiting branch-and-bound algorithm for mixed-integer model predictive control. In *2019 18th European Control Conference (ECC)*, pages 2763–2768. IEEE, 2019.
- [44] Jean-Baptiste Hiriart-Urruty and Claude Lemaréchal. *Convex analysis and minimization algorithms I: Fundamentals*, volume 305. Springer science & business media, 2013.
- [45] Reiner Horst and Hoang Tuy. *Global optimization: Deterministic approaches*. Springer Science & Business Media, 2013.

- [46] Daniel Ioan, Ionela Prodan, Sorin Olaru, Florin Stoican, and Silviu-Iulian Niculescu. Mixed-integer programming in motion planning. *Annual Reviews in Control*, 2020.
- [47] Richard M Karp. Reducibility among combinatorial problems. In *Complexity of computer computations*, pages 85–103. Springer, 1972.
- [48] Bachir El Khadir, Jean Bernard Lasserre, and Vikas Sindhvani. Piecewise-linear motion planning amidst static, moving, or morphing obstacles. *arXiv preprint arXiv:2010.08167*, 2020.
- [49] Jongrae Kim and Joao P Hespanha. Discrete approximations to continuous shortest-path: Application to minimum-risk path planning for groups of UAVs. In *42nd IEEE International Conference on Decision and Control (IEEE Cat. No. 03CH37475)*, volume 2, pages 1734–1740. IEEE, 2003.
- [50] Benoit Landry, Robin Deits, Peter R Florence, and Russ Tedrake. Aggressive quadrotor flight through cluttered environments using mixed integer programming. In *2016 IEEE international conference on robotics and automation (ICRA)*, pages 1469–1475. IEEE, 2016.
- [51] Monique Laurent. A comparison of the Sherali-Adams, Lovász-Schrijver, and Lasserre relaxations for 0–1 programming. *Mathematics of Operations Research*, 28(3):470–496, 2003.
- [52] Der-Tsai Lee and Franco P Preparata. Euclidean shortest paths in the presence of rectilinear barriers. *Networks*, 14(3):393–410, 1984.
- [53] Fajie Li and Reinhard Klette. *Euclidean shortest paths*. Springer, 2011.
- [54] Guangzhi Li and Rahul Simha. The partition coloring problem and its application to wavelength routing and assignment. In *Proceedings of the First Workshop on Optical Networks*, volume 1. Citeseer, 2000.
- [55] Wen-Jui Li, H-S Jacob Tsao, and Osman Ulular. The shortest path with at most n nodes in each of the series/parallel clusters. *Networks*, 26(4):263–271, 1995.
- [56] Jiaming Liang, Stefano Di Cairano, and Rien Quirynen. Early termination of convex QP solvers in mixed-integer programming for real-time decision making. *IEEE Control Systems Letters*, 2020.
- [57] László Lovász and Alexander Schrijver. Cones of matrices and set-functions and 0–1 optimization. *SIAM journal on optimization*, 1(2):166–190, 1991.
- [58] Tomás Lozano-Pérez and Michael A Wesley. An algorithm for planning collision-free paths among polyhedral obstacles. *Communications of the ACM*, 22(10):560–570, 1979.
- [59] Tobia Marcucci. Shortest paths in graphs of convex sets: supporting software. URL: <https://github.com/TobiaMarcucci/shortest-paths-in-graphs-of-convex-sets>.
- [60] Tobia Marcucci, Robin Deits, Marco Gabiccini, Antonio Bicchi, and Russ Tedrake. Approximate hybrid model predictive control for multi-contact push recovery in complex environments. In *2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)*, pages 31–38. IEEE, 2017.
- [61] Tobia Marcucci and Russ Tedrake. Mixed-integer formulations for optimal control of piecewise-affine systems. In *Proceedings of the 22nd ACM International Conference on Hybrid Systems: Computation and Control*, pages 230–239, 2019.

- [62] Tobia Marcucci and Russ Tedrake. Warm start of mixed-integer programs for model predictive control of hybrid systems. *IEEE Transactions on Automatic Control*, 66(6):2433–2448, 2021.
- [63] Marcos ROA Maximo and Rubens JM Afonso. Mixed-integer quadratic programming for automatic walking footstep placement, duration, and rotation. *Optimal Control Applications and Methods*, 2020.
- [64] David Q Mayne, James B Rawlings, Christopher V Rao, and Pierre OM Scokaert. Constrained model predictive control: Stability and optimality. *Automatica*, 36(6):789–814, 2000.
- [65] Garth P McCormick. Computability of global solutions to factorable nonconvex programs: Part i—convex underestimating problems. *Mathematical programming*, 10(1):147–175, 1976.
- [66] Hanna Michalska and David Q Mayne. Robust receding horizon control of constrained nonlinear systems. *IEEE Transactions on Automatic Control*, 38(11):1623–1633, 1993.
- [67] Clair E Miller, Albert W Tucker, and Richard A Zemlin. Integer programming formulation of traveling salesman problems. *Journal of the ACM (JACM)*, 7(4):326–329, 1960.
- [68] Nicholas Moehle and Stephen Boyd. A perspective-based convex relaxation for switched-affine optimal control. *Systems & Control Letters*, 86:34–40, 2015.
- [69] MOSEK ApS. *The MOSEK Optimizer API for C. Version 9.2.*, 2021. URL: <http://docs.mosek.com/9.2/capi/index.html>
- [70] Young-Soo Myung, Chang-Ho Lee, and Dong-Wan Tcha. On the generalized minimum spanning tree problem. *Networks*, 26(4):231–241, 1995.
- [71] Vihangkumar V Naik and Alberto Bemporad. Embedded mixed-integer quadratic optimization using accelerated dual gradient projection. *IFAC-PapersOnLine*, 50(1):10723–10728, 2017.
- [72] Charles E Noon and James C Bean. An efficient transformation of the generalized traveling salesman problem. *INFOR: Information Systems and Operational Research*, 31(1):39–44, 1993.
- [73] Simeon Ntafos. Watchman routes under limited visibility. *Computational Geometry*, 1(3):149–170, 1992.
- [74] Christos H Papadimitriou. An algorithm for shortest-path motion in three dimensions. *Information processing letters*, 20(5):259–263, 1985.
- [75] Petrica C Pop. *Generalized network design problems: Modeling and optimization*, volume 1. Walter de Gruyter, 2012.
- [76] Petrică C Pop. The generalized minimum spanning tree problem: An overview of formulations, solution procedures and latest advances. *European Journal of Operational Research*, 283(1):1–15, 2020.
- [77] Petrică C Pop, Oliviu Matei, and Corina Pop Sitar. An improved hybrid algorithm for solving the generalized vehicle routing problem. *Neurocomputing*, 109:76–83, 2013.
- [78] Arthur Richards and Jonathan How. Mixed-integer programming for control. In *Proceedings of the 2005, American Control Conference, 2005.*, pages 2676–2683. IEEE, 2005.

- [79] Arthur Richards and Jonathan P How. Aircraft trajectory planning with collision avoidance using mixed integer linear programming. In *Proceedings of the 2002 American Control Conference (IEEE Cat. No. CH37301)*, volume 3, pages 1936–1941. IEEE, 2002.
- [80] Arthur Richards and Jonathan P How. Robust variable horizon model predictive control for vehicle maneuvering. *International Journal of Robust and Nonlinear Control*, 16(7):333–351, 2006.
- [81] R Tyrrell Rockafellar. *Convex analysis*. Number 28. Princeton university press, 1970.
- [82] Alexander Schrijver. *Combinatorial optimization: polyhedra and efficiency*, volume 24. Springer Science & Business Media, 2003.
- [83] Pierre OM Scokaert and David Q Mayne. Min-max feedback model predictive control for constrained linear systems. *IEEE Transactions on Automatic control*, 43(8):1136–1142, 1998.
- [84] Hanif D Sherali and Warren P Adams. A hierarchy of relaxations between the continuous and convex hull representations for zero-one programming problems. *SIAM Journal on Discrete Mathematics*, 3(3):411–430, 1990.
- [85] Hanif D Sherali and Warren P Adams. *A reformulation-linearization technique for solving discrete and continuous nonconvex problems*, volume 31. Springer Science & Business Media, 2013.
- [86] Hanif D Sherali and Amine Alameddine. A new reformulation-linearization technique for bilinear programming problems. *Journal of Global optimization*, 2(4):379–410, 1992.
- [87] Hanif D Sherali and Leo Liberti. Reformulation-linearization technique for global optimization. *Encyclopedia of Optimization*, 2:3263–3268, 2009.
- [88] Eduardo Sontag. Nonlinear regulation: The piecewise linear approach. *IEEE Transactions on automatic control*, 26(2):346–358, 1981.
- [89] Bartolomeo Stellato, Vihangkumar V Naik, Alberto Bemporad, Paul Goulart, and Stephen Boyd. Embedded mixed-integer quadratic optimization using the OSQP solver. In *2018 European Control Conference (ECC)*, pages 1536–1541. IEEE, 2018.
- [90] Russ Tedrake and the Drake Development Team. Drake: Model-based design and verification for robotics, 2019. URL: <https://drake.mit.edu>.
- [91] John N Tsitsiklis. Efficient algorithms for globally optimal trajectories. *IEEE Transactions on Automatic Control*, 40(9):1528–1538, 1995.
- [92] Michael Vitus, Vijay Pradeep, Gabriel Hoffmann, Steven Waslander, and Claire Tomlin. Tunnel-MILP: Path planning with sequential convex polytopes. In *AIAA guidance, navigation and control conference and exhibit*, page 7132, 2008.
- [93] Andreas Wächter and Lorenz T Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical programming*, 106(1):25–57, 2006.
- [94] Chin Wei-Pang and Simeon Ntafos. The zookeeper route problem. *Information Sciences*, 63(3):245–259, 1992.

- [95] David P Williamson and David B Shmoys. *The design of approximation algorithms*. Cambridge university press, 2011.
- [96] Yang Yang, Mingen Lin, Jinhui Xu, and Yulai Xie. Minimum spanning tree with neighborhoods. In *International Conference on Algorithmic Applications in Management*, pages 306–316. Springer, 2007.

A Alternative Mixed-Integer Convex Formulations

In Section 5 we have proposed a technique to reformulate the bilinear program (10) as an MICP. However, multiple alternative approaches could be used to achieve this result. In this appendix we present two natural alternatives to the proposed formulation, and we provide numerical evidence of why, in general, the MICP (21) is to be preferred.

A.1 Edge-by-Edge Formulation

The simplest approach to derive an MICP formulation of the SPP (1) is to analyze what the constraints of the bilinear program (10) imply for each edge independently. This in contrast with Section 5.2 where we grouped all the edges that share a common vertex v , and we derived our MICP leveraging the structure of the polytopes Φ_v .

Zooming in on the edge $e = (u, v) \in E$, in case of a binary flow $\varphi_e \in \{0, 1\}$, the constraints in (10) yield a disjunction between two convex sets:

$$\{(x_u, x_v, y_e, z_e, \varphi_e) : \varphi_e = 0, x_u \in X_u, x_v \in X_v, y_e = z_e = 0\}, \quad (56a)$$

$$\{(x_u, x_v, y_e, z_e, \varphi_e) : \varphi_e = 1, y_e = x_u \in X_u, z_e = x_v \in X_v\}. \quad (56b)$$

This discrete choice could be easily encoded using the big-M method, but this technique is well known to yield very loose convex relaxations. The strongest possible mixed-integer convex formulation of the disjunction (56) is obtained via the convex-hull method [13, 3] and, using the set-perspective notation, it reads

$$(y_e, \varphi_e) \in \tilde{X}_u, \quad (x_u - y_e, 1 - \varphi_e) \in \tilde{X}_u, \quad (57a)$$

$$(z_e, \varphi_e) \in \tilde{X}_v, \quad (x_v - z_e, 1 - \varphi_e) \in \tilde{X}_v. \quad (57b)$$

Note that when $\varphi_e = 0$ these conditions simplify to (56a), while for $\varphi_e = 1$ they give us back (56b). The MICP corresponding to this formulation is obtained by substituting the inclusion $x_v \in X_v$ in (10b) and the bilinear constraint (10c) with (57), and by requiring the flow variables to be binary.

Numerical evidence of the higher performance of the MICP (21) with respect to the one we just discussed is given in Section A.3. For the moment let us notice that this edge-by-edge formulation can also be obtained as a special case of the technique we presented in Section 7. If, instead of using the local flow polytope Φ_v in the definition (26) of the set Ω_v , we just use the trivial bound $\varphi_v \in [0, 1]^{|E_v|}$, the relaxation Ω'_v from (30) gives us exactly the constraints (57). Therefore, we expect the MICP (21) to be much stronger than the one obtained here: in fact, as seen in Section 7.2.1, tighter bounds on the value of φ_v yield tighter relaxations Ω'_v . As a visual example, for a vertex $v \notin \{s, t\}$ with two incoming edges and one outgoing edge, Figure 11 compares the polytope Φ_v and the cube $[0, 1]^{|E_v|}$.

Finally, we notice that this edge-by-edge formulation is slightly larger than the MICP (21); this because the variables x_v cannot be removed from the problem formulation as done in Section 5.3. Here we have $|E|$ binary variables and $d|V| + 2d|E|$ continuous variables. The number of constraints is $2|V| + 4h(d)|E|$, where $h(d)$ is defined at the bottom of Section (5.3).

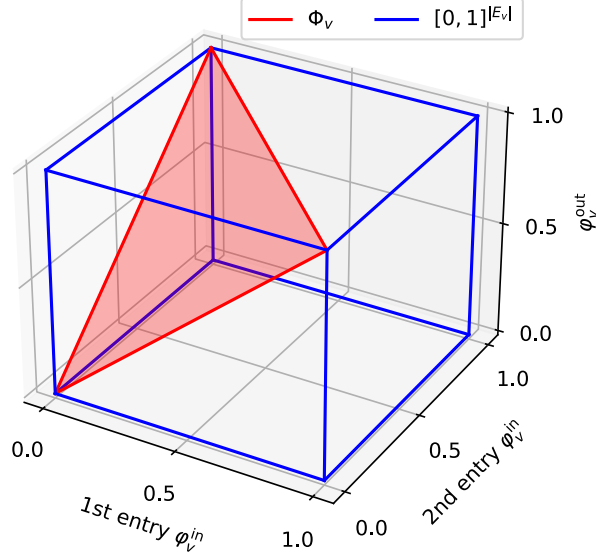


Figure 11: The local flow polytope Φ_v from (4) and the unit cube $[0, 1]^{|E_v|}$ for a vertex $v \notin \{s, t\}$ with two incoming edges and one outgoing edge. The flow vector φ_v is decomposed as $(\varphi_v^{\text{in}}, \varphi_v^{\text{out}})$, where φ_v^{in} collects the two incoming flows and φ_v^{out} is the outgoing one. Grouping the edges that share a common vertex v , instead of operating on each edge individually, allows us to exploit much tighter bounds on the values of the flows (Φ_v instead of $[0, 1]^{|E_v|}$), and to design a much stronger MICP.

A.2 Convex-Hull Formulation

An alternative mixed-integer convex formulation of the feasible set of problem (10) has been mentioned at the end of Section 7.3. This consists in a direct computation of the convex hulls of the sets $\Omega_v \cap C_v$, where $C_v := \{(\varphi, x, M) : \varphi \in \text{ext}(\Phi_v)\}$. The derivation of the following families of constraints is a straightforward but tedious application of the convex-hull method [13, 3], here we only report their final descriptions.

For the source, the convex hull of $\Omega_s \cap C_s$ can be described without the introduction of auxiliary variables via the following constraints:

$$(x_s, 1) = \sum_{e \in E_s^{\text{out}}} (y_e, \varphi_e), \quad (58a)$$

$$\varphi_e = 0, \quad z_e = 0, \quad \forall e \in E_s^{\text{in}}, \quad (58b)$$

$$(y_e, \varphi_e) \in \tilde{X}_s, \quad \forall e \in E_s^{\text{out}}. \quad (58c)$$

In a specular manner, the convex hull of set $\Omega_t \cap C_t$ is delimited by the constraints

$$(x_t, 1) = \sum_{e \in E_t^{\text{in}}} (z_e, \varphi_e), \quad (59a)$$

$$(z_e, \varphi_e) \in \tilde{X}_t, \quad \forall e \in E_t^{\text{in}}, \quad (59b)$$

$$\varphi_e = 0, \quad y_e = 0, \quad \forall e \in E_t^{\text{out}}. \quad (59c)$$

The description of the convex hull of $\Omega_v \cap C_v$ when $v \notin \{s, t\}$ is more involved, since we need to consider all the possible combinations of incoming and outgoing edges. In this regard, we introduce

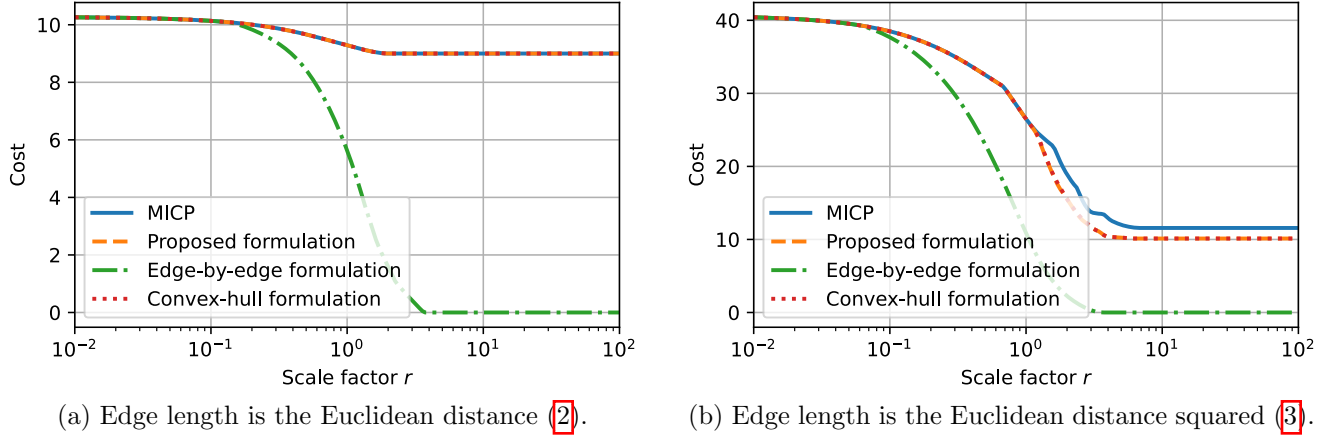


Figure 12: Application of the formulations from Appendix A to the numerical example from Section 11.1. The two plots compare the cost of the convex relaxations of these formulations with the optimal value of the SPP for different sizes r of the sets X_v . As expected the edge-by-edge formulation is extremely weak, whereas the convex-hull formulation turns out to be as strong as the proposed MICP (21), despite its substantially larger size.

the auxiliary variables $\varphi_{io} \in \mathbb{R}$ and $x_{io} \in \mathbb{R}^d$ for all edges $i \in E_v^{\text{in}}$ and $o \in E_v^{\text{out}}$. The variable φ_{io} can be interpreted as the units of flow that enter vertex v through the edge i and leave it through o . The variable x_{io} must match x_v when $\varphi_{io} = 1$ and collapse to zero when $\varphi_{io} = 0$. Applying the convex-hull method, after several manipulations, we arrive to

$$\begin{bmatrix} x_v - \sum_{i \in E_v^{\text{in}}, o \in E_v^{\text{out}}} x_{io} \\ 1 - \sum_{i \in E_v^{\text{in}}, o \in E_v^{\text{out}}} \varphi_{io} \end{bmatrix} \in \tilde{X}_v, \quad (60a)$$

$$(z_e, \varphi_e) = \sum_{o \in E_v^{\text{out}}} (x_{eo}, \varphi_{eo}), \quad \forall e \in E_v^{\text{in}}, \quad (60b)$$

$$(y_e, \varphi_e) = \sum_{i \in E_v^{\text{in}}} (x_{ie}, \varphi_{ie}), \quad \forall e \in E_v^{\text{out}}, \quad (60c)$$

$$(x_{io}, \varphi_{io}) \in \tilde{X}_v, \quad \forall i \in E_v^{\text{in}}, o \in E_v^{\text{out}}. \quad (60d)$$

The MICP corresponding to this formulation is obtained by minimizing (10a) subject to constraints (58), (59), (60) as well as the binary requirements $\varphi_e \in \{0, 1\}$ for all $e \in E_s \cup E_t$ and $\varphi_{io} \in \{0, 1\}$ for all $i \in E_v^{\text{in}}, o \in E_v^{\text{out}}, v \in V - \{s, t\}$. Regarding the size of this MICP we only mention that the number of constraints, binary variables, and continuous variables is now proportional to the sum of the products of the indegrees and the outdegrees of the vertices in the graph. As we see in the following subsection, this is a burden that strongly limits the performance of this formulation.

A.3 Numerical Comparison

We present a numerical comparison of the MICP (21) with the two formulations discussed in this appendix. We start from the two-dimensional example described in Section 11.1. In Figure 12 we report the curves corresponding to Figure 6 for the two formulations described above. As expected, the edge-by-edge formulation from Appendix A.1 is extremely weak, and with both the edge lengths (2) and (3), the optimal value of its convex relaxation converges to zero as the sets X_v grow in size. Notice

that this formulation does not recover the simple lower bound from Proposition 5. On the other hand, the performance of the convex-hull formulation from Appendix A.2 is indistinguishable from the one of the proposed MICP (21), despite the significantly larger programs that this method requires to solve.

We then consider the optimal-control problem described in Section 11.3. The edge-by-edge formulation has a relaxation gap of 91%, and the corresponding MICP is solved in 10.3 s.¹² This outperforms the formulation from [68, 61] but is still an order of magnitude slower than the proposed MICP (21). The convex-hull formulation has a relaxation gap of 18% which, as expected, is smaller than the one of the MICP (21). However, the large size of this formulation makes the MICP solution time extremely high: 588 s.

Finally we report that even the convex-hull formulation does not overcome the issues highlighted in Section 11.4. For both the examples its convex relaxation gives the same solution as the MICP (21).

B Proofs

We gather in this appendix the proofs whose content is not strictly relevant to the discussion in the main body of the paper.

B.1 Sketch of Proof of Theorem 2

The construction from [12, Theorem 2.3.2] allows to reduce the 3-SAT problem, whose NP-completeness is well known [47], to the SPP (1) in polynomial time. The idea is to stack multiple layers of two-dimensional convex sets X_v in a three-dimensional space. With the source X_s on top of the stack and the target X_t at the bottom, the sets X_v are designed so that there are exponentially many s - t shortest paths, one per assignment of the variables in the 3-SAT formula. Paths associates with infeasible assignments can be then bent and filtered out by forcing them to traverse suitable convex sets.

All the “substructures” needed to construct the three-dimensional environment from [12, Section 2.2] can be easily described in terms of convex sets X_v instead of “plates,” “slits,” and “barriers.” Importantly, none of these substructures requires a set X_v to be unbounded. The stacked structure of this environment guarantees that a shortest path traverses the plates, and hence our convex sets, in a sequential manner. This ensures the absence of cycles in our edge set E . As for the Euclidean SPP, the resulting instance of the SPP (1) has size polynomial in the number of variables and clauses in the 3-SAT formula, and its optimal value equals a known constant if and only if the formula is satisfiable.

B.2 Proof of the Claims from Section 4.1

We verify that the points listed in Section 4.1 are actually the extreme points of the local flow polytopes Φ_v . For the source vertex s and the target t the claim is verified very easily. Below we focus on the case $v \notin \{s, t\}$.

To show that $\text{ext}(\Phi_v) = \{0_{|E_v|}\} \cup (\Delta_{|E_v^{\text{in}}|} \times \Delta_{|E_v^{\text{out}}|}) =: \hat{\Phi}_v$, we verify that $\Phi_v = \text{conv}(\hat{\Phi}_v)$. Since the three conditions in (4) are verified by all the points in $\hat{\Phi}_v$, we have $\text{conv}(\hat{\Phi}_v) \subseteq \Phi_v$. For the reverse inclusion, we explicitly decompose a point $\varphi_v \in \Phi_v$ as a convex combination of the elements in $\hat{\Phi}_v$. The decomposition is trivial if $\varphi_v = 0$, we then let $\varphi_v \neq 0$. To the point in $\hat{\Phi}_v$ associated with a unit of flow traversing the edges $i \in E_v^{\text{in}}$ and $o \in E_v^{\text{out}}$ we assign the coefficient $\alpha_{io} := \varphi_i \varphi_o / \sum_{e \in E_v^{\text{in}}} \varphi_e$, while we pair the zero vector in $\hat{\Phi}_v$ with $\alpha_0 := 1 - \sum_{i \in E_v^{\text{in}}, o \in E_v^{\text{out}}} \alpha_{io}$. These coefficients define a valid convex

¹²The result returned by Mosek with default settings was inaccurate for this problem. This issue was fixed by setting the parameter MSK_IPAR_INTPNT_SOLVE_FORM to MSK_SOLVE_PRIMAL.

combination: they sum up to one, the nonnegativity of the flows (4a) implies $\alpha_{io} \geq 0$, and the degree constraint (4c) ensures that

$$\alpha_0 := 1 - \frac{\sum_{i \in E_v^{\text{in}}, o \in E_v^{\text{out}}} \varphi_i \varphi_o}{\sum_{e \in E_v^{\text{in}}} \varphi_e} = 1 - \frac{\left(\sum_{i \in E_v^{\text{in}}} \varphi_i\right) \left(\sum_{o \in E_v^{\text{out}}} \varphi_o\right)}{\sum_{e \in E_v^{\text{in}}} \varphi_e} = 1 - \sum_{o \in E_v^{\text{out}}} \varphi_o \geq 0.$$

We are left to check that combining with these coefficients the elements of $\hat{\Phi}_v$ we actually get φ_v . The entry corresponding to the edge $i \in E_v^{\text{in}}$ of this combination is

$$\sum_{o \in E_v^{\text{out}}} \alpha_{io} = \frac{\sum_{o \in E_v^{\text{out}}} \varphi_i \varphi_o}{\sum_{e \in E_v^{\text{in}}} \varphi_e} = \varphi_i \frac{\sum_{o \in E_v^{\text{out}}} \varphi_o}{\sum_{e \in E_v^{\text{in}}} \varphi_e} = \varphi_i,$$

where the last equality uses the conservation of flow (4b). Similarly, the entry corresponding to the edge $o \in E_v^{\text{out}}$ is correctly set to $\sum_{i \in E_v^{\text{in}}} \alpha_{io} = \varphi_o$.

B.3 Proof of Proposition 3

We show mutual inclusion. The inclusion $\Omega \cap C \subseteq \Omega$ remains true if we take the convex hull of the two sets. For the other direction it suffices to show that $\Omega \subseteq \text{conv}(\Omega \cap C)$. Let $(\varphi, x, M) \in \Omega$. Listing the extreme points of Φ as $\text{ext}(\Phi) = \{\hat{\varphi}_j\}_{j \in J}$, we consider nonnegative scalars α_j such that $\sum_{j \in J} \alpha_j = 1$ and $\sum_{j \in J} \alpha_j \hat{\varphi}_j = \varphi$. We define $M_j := x \hat{\varphi}_j^\top$ and we take a convex combination of the points $(\hat{\varphi}_j, x, M_j)$ with coefficients α_j . These coefficients define a valid convex combination. For all $j \in J$, the inclusion $(\hat{\varphi}_j, x, M_j) \in \Omega \cap C$ follows from $\hat{\varphi}_j \in \text{ext}(\Phi)$, $x \in X$, and the definition of M_j . Finally, since $\sum_{j \in J} \alpha_j M_j = \sum_{j \in J} \alpha_j (x \hat{\varphi}_j^\top) = x \hat{\varphi}^\top = M$, we have $\sum_{j \in J} \alpha_j (\hat{\varphi}_j, x, M_j) = (\varphi, x, M)$. Thus $(\varphi, x, M) \in \text{conv}(\Omega \cap C)$.

B.4 Proof of Proposition 4

Assume we are given a set of variables $\{\varphi_e, y_e, z_e\}_{e \in E}$ that verify the constraints of the convex relaxation of the MICP (21). Similarly, let $\{p_v, q_v, r_v\}_{v \in V}$ and $\{a_e, b_e, \alpha_e, \beta_e\}_{e \in E}$ verify the constraints of the dual program (37). We need to show that difference between the primal and the dual objectives is nonnegative:

$$\sum_{e \in E} \tilde{\ell}_e(y_e, z_e, \varphi_e) - p_s + p_t + \sum_{v \in V - \{t\}} q_v \geq 0. \quad (61)$$

We start by using Fenchel-Young inequality: $f^*(a) \geq a^\top x - f(x)$ for all x and a . Applying this inequality for the function $\tilde{\ell}_e$, with $e = (u, v)$, we have

$$\begin{aligned} \tilde{\ell}_e(y_e, z_e, \varphi_e) &\geq (r_u + a_e)^\top y_e + (-r_v + \alpha_e)^\top z_e + (p_u - p_v - q_u + b_e + \beta_e) \varphi_e \\ &\quad - (\tilde{\ell}_e)^*(r_u + a_e, -r_v + \alpha_e, p_u - p_v - q_u + b_e + \beta_e). \end{aligned} \quad (62)$$

Following [16, Proposition 2.3(iv)], the conjugate of the perspective of ℓ_e is seen to be the indicator function of the set $T_e := \{(a, b, c) : \ell_e^*(a, b) + c \leq 0\}$, i.e.: $(\tilde{\ell}_e)^* = \iota_{T_e}$. Evaluating this indicator function at the point where $(\tilde{\ell}_e)^*$ is evaluated in (62) we get zero: in fact, these dual variables are forced by the constraint (37b) to lie in the set T_e . In addition, we notice that the terms $a_e^\top y_e + b_e \varphi_e$ and $\alpha_e^\top z_e + \beta_e \varphi_e$ in (62) are nonnegative. To see this for the first term, note that the constraints (21b) and (37c) enforce

$(y_e, \varphi_e) \in \tilde{X}_u$ and $(a_e, b_e) \in X_u^\circ$, and recall that $\tilde{X}_u$ and X_u° are dual cones by Lemma 3(b). The nonnegativity of the second term is shown similarly.

The observations above show that the left-hand side in (61) is lower bounded by

$$\sum_{e=(u,v) \in E} \left(r_u^\top y_e - r_v^\top z_e + (p_u - p_v - q_u) \varphi_e \right) - p_s + p_t + \sum_{v \in V - \{t\}} q_v.$$

We rearrange the summation over the edges, and we rewrite our lower bound as

$$\sum_{v \in V} \left(\sum_{e \in E_v^{\text{out}}} \left(r_v^\top y_e + (p_v - q_v) \varphi_e \right) - \sum_{e \in E_v^{\text{in}}} \left(r_v^\top z_e + p_v \varphi_e \right) \right) - p_s + p_t + \sum_{v \in V - \{t\}} q_v.$$

To conclude the proof, we show that the latter expression can be decomposed as the sum of nonnegative terms. Collecting the terms involving the multipliers p_v of the conservations of flow, we have

$$\sum_{v \in V} \left(\sum_{e \in E_v^{\text{out}}} p_v \varphi_e - \sum_{e \in E_v^{\text{in}}} p_v \varphi_e \right) - p_s + p_t = \sum_{v \in V} p_v \left(\sum_{e \in E_v^{\text{out}}} \varphi_e + \delta_{tv} - \sum_{e \in E_v^{\text{in}}} \varphi_e - \delta_{sv} \right) = 0,$$

where the second equality uses the conservation of flow in (21c). We proceed similarly for the terms involving multipliers q_v of the degree constraints:

$$- \sum_{v \in V} \sum_{e \in E_v^{\text{out}}} q_v \varphi_e + \sum_{v \in V - \{t\}} q_v = \sum_{v \in V} q_v \left(1 - \delta_{tv} - \sum_{e \in E_v^{\text{out}}} \varphi_e \right) \geq 0,$$

where the inequality uses the degree constraint in (21c) and the nonnegativity of q_v from (37d). Finally, we are left with the terms involving the multipliers r_v of the spatial conservations of flow:

$$\sum_{v \in V} \left(\sum_{e \in E_v^{\text{out}}} r_v^\top y_e - \sum_{e \in E_v^{\text{in}}} r_v^\top z_e \right) = \sum_{v \in V} r_v^\top \left(\sum_{e \in E_v^{\text{out}}} y_e - \sum_{e \in E_v^{\text{in}}} z_e \right) = 0,$$

where the second equality uses the spatial conservation of flow (21d) and the dual constraints (37e).

B.5 Proof of Proposition 5

We start by specializing the dual program (37) to the edge length (38). In this case, the conjugate function in the dual constraint (37b) becomes

$$\ell_e^*(r_u + a_e, -r_v + c_e) = \sup_{x_u, x_v} \left((r_u + a_e)^\top x_u + (-r_v + c_e)^\top x_v - \ell(x_v - x_u) \right). \quad (63)$$

When $r_u + a_e - r_v + c_e \neq 0$ this supremum is infinite and the dual problem is infeasible (this is seen by setting $x_u = x_v$ and recalling that $\ell(0) = 0$). We then have a “hidden” dual constraint

$$r_u + a_e = r_v - c_e, \quad \forall e = (u, v) \in E. \quad (64)$$

Using this, the conjugate (63) becomes

$$\sup_{x_u, x_v} \left((-r_v + c_e)^\top (x_v - x_u) - \ell(x_v - x_u) \right) = \ell^*(-r_v + c_e),$$

and constraint (37b) reads

$$p_u - p_v - q_u + b_e + \beta_e \leq -\ell^*(-r_v + \alpha_e), \quad \forall e = (u, v) \in E. \quad (65)$$

We have now the tools to prove Proposition 5: the plan is to synthesize a dual feasible solution whose cost coincides with the optimal value of (40). The thesis is then implied by weak duality.

We start by deriving the dual of problem (40). Using the indicator functions ι_{X_s} and ι_{X_t} , we reformulate this program as the minimization of a convex function¹³ subject to linear constraints only:

$$\begin{aligned} & \text{minimize} && (|V| - 1)\ell(x) + \iota_{X_s}(x_s) + \iota_{X_t}(x_t) \\ & \text{subject to} && (|V| - 1)x = x_t - x_s. \end{aligned}$$

Here the decision variables are x , x_s , and x_t . The dual of the latter minimization is easily derived using conjugate functions (see [9, Section 5.1.6]) and it reads

$$\text{maximize} \quad -(|V| - 1)\ell^*(-r) - \sigma_{X_s}(-r) - \sigma_{X_t}(r). \quad (66)$$

Here σ_{X_s} and σ_{X_t} are support functions¹⁴ and the only decision variable is r . Under mild assumptions on the edge length ℓ and the convex sets X_s and X_t , strong duality holds for the pair (40) and (66). The optimal values of these programs are hence equal. The plan is then to find a (partial) feasible assignment for the variables of the dual (37) that reduces this program to (66).

We set the following values for the dual variables:

- Spatial conservation of flow (21d): $r_v := (1 - \delta_{sv} - \delta_{tv})r$ for all $v \in V$, where r is a decision variable.
- Nonnegativity constraint (21b): $a_e := (\delta_{su} + \delta_{tu})r$, $b_e := (\delta_{su} + \delta_{tu})\sigma_{X_u}(-r)$, $\alpha_e := -(\delta_{sv} + \delta_{tv})r$, and $\beta_e := (\delta_{sv} + \delta_{tv})\sigma_{X_v}(r)$ for all $e = (u, v) \in E$.
- Conservation of flow in (21c): $p_v := (\delta_{sv} + \delta_{tv})\sigma_{X_v}(r)$ for all $v \in V$.
- Degree constraint in (21c): $q_v := \ell^*(-r) + (\delta_{sv} + \delta_{tv})(\sigma_{X_v}(r) + \sigma_{X_v}(-r))$ for all $v \in V$.

With this assignment the dual objective (37a) simplifies exactly to (66). We are then left to check that all the dual constraints are verified:

- For the nonnegativity (37d) of q_v we note that: the assumption $\ell(0) = 0$ implies $\ell^* \geq 0$, the term $\delta_{sv} + \delta_{tv}$ is clearly nonnegative, and the term $\sigma_{X_v}(r) + \sigma_{X_v}(-r)$ is equal to $\sup_{x \in X_v} r^\top x - \inf_{x \in X_v} r^\top x$, which is also nonnegative.
- The hidden dual equality (64) and the potential jump (65) are verified substituting the given multipliers and simplifying.
- The first constraint in (37c) requires that $a_e^\top x + b_e \geq 0$ for all $x \in X_u$. After substituting, we get the condition $(\delta_{su} + \delta_{tu})(r^\top x + \sigma_{X_u}(-r)) \geq 0$ for all $x \in X_u$: the factor $\delta_{su} + \delta_{tu}$ is nonnegative and, using the definition of σ_{X_u} , we see that the same is true also for the second factor. Similarly, the second constraint in (37c) becomes $(\delta_{sv} + \delta_{tv})(-r^\top x + \sigma_{X_v}(r)) \geq 0$ for all $x \in X_v$, and it is easily verified to hold using the definition of σ_{X_v} .

¹³Recall that the indicator function of a convex set is a convex function.

¹⁴For a set $S \subseteq \mathbb{R}^n$, the *support function* is defined as $\sigma_S(a) := \sup_{x \in S} (a^\top x)$. Note that the support function is the conjugate of the indicator function ($\sigma_S = \iota_S^*$), thus it is always convex.