Approximating optimization problems in graphs with locational uncertainty

Marin Bougeret[1]   Jérémy Omer[2]   Michael Poss[1]

[1]   *LIRMM, University of Montpellier, CNRS, Montpellier, France*
[2]   *IRMAR, INSA Rennes, Rennes, France*

Many combinatorial optimization problems can be formulated as the search for a subgraph that satisfies certain properties and minimizes the total weight. We assume here that the vertices correspond to points in a metric space and can take any position in given uncertainty sets. Then, the cost function to be minimized is the sum of the distances for the worst positions of the vertices in their uncertainty sets. We propose two types of polynomial-time approximation algorithms. The first one relies on solving a deterministic counterpart of the problem where the uncertain distances are replaced with maximum pairwise distances. We study in details the resulting approximation ratio, which depends on the structure of the feasible subgraphs and whether the metric space is Ptolemaic or not. The second algorithm is a fully-polynomial time approximation scheme for the special case of $s - t$ paths.

Keywords: robust optimization, approximation algorithms, dynamic programming

1   Introduction

Given a graph $\mathbb{G} = (\mathbb{V}, \mathbb{E})$ and a family of feasible subgraphs $\mathcal{G}$ of $\mathbb{G}$, many discrete optimization problems amount to find the cheapest subgraph in $\mathcal{G}$. In this paper, we assume that $\mathbb{G}$ is a graph embedded into a given metric space $(\mathcal{M}, d)$ so that every vertex $i \in \mathbb{V}$ is associated to a point $u_i \in \mathcal{M}$ and the weight of edge $\{i, j\} \in \mathbb{E}$ is equal to the distance $d(u_i, u_j)$. Then, the cost of a subgraph $G \in \mathcal{G}$ is defined as the sum of the weights of its edges. Denoting by $E[G]$ the set of edges of G, the cost of graph $G \in \mathcal{G}$ is given by

$$c_d(u, G) = \sum_{\{i,j\} \in E[G]} d(u_i, u_j),$$

leading to the combinatorial optimization problem

$$\min_{G \in \mathcal{G}} \sum_{\{i,j\} \in E[G]} d(u_i, u_j). \tag{Π}$$

The subscript d on c_d will be removed when clear from the context, denoting the cost by $c(u, G)$.

An example of problem Π arises in data clustering, where one wishes to partition a given set of data points $\{u_1, \ldots, u_n\} \subset \mathcal{M}$ into at most K sets so as to minimize the sum of all dissimilarities between

points that belong to the same element of the partition. In this example, $\mathbb{G}$ is a complete graph, every vertex i of $\mathbb{V}$ represents one of the data point u_i, and $\mathcal{G}$ consists of all disjoint unions of cliques covering $\mathbb{G}$. Furthermore, the distance $d(u_i, u_j)$ between any two vertices $i, j \in \mathbb{V}$ measures the dissimilarity between the corresponding data points. This dissimilarity may involve Euclidean distances, for instance when comparing numerical values, as well as more ad-hoc distances when comparing ordinal values instead. Problem Π encompasses other applications, such as subway network design or facility location. In the the network design problem, one wishes to find a cheapest feasible subgraph of $\mathbb{G}$ connecting a certain number of vertices. Thus, $\mathcal{G}$ consists of Steiner trees covering a given set of terminals and the distance used is typically Euclidean, see Gutiérrez-Jarpa et al. (2013). In facility location problems, set $\mathcal{G}$ consists of unions of disjoint stars, the centers and the leaves of the stars respectively representing the facility and the clients. In that example, the distance typically involves shortest paths in an underlying road network represented by an auxiliary weighted graph $G_{\mathcal{M}} = (V_{\mathcal{M}}, E_{\mathcal{M}})$, leading to a graph-induced metric space, see Melkote and Daskin (2001).

More formally, we are interested in the class $\mathcal{S}$ of deterministic combinatorial optimization problems that can be formulated as Π. Any $\Pi \in \mathcal{S}$ represents a specific problem, such as the shortest path or the minimum spanning tree. An instance of problem Π is defined by its input $I = (\mathbb{G}, \alpha, u, d)$, where $\mathbb{G}$ is an undirected simple graph, α contains problem-specific additional inputs, and d is the distance matrix of the set $\bigcup_{i \in \mathbb{V}} \{u_i\}$. We denote by $\mathcal{G}(I)$ the set of all subgraphs (not necessarily induced) of $\mathbb{G}$ that satisfy the constraints specific to Π for input I. For instance, for the Shortest Path problem (SP), the additional input $\alpha = \{s, t\}$ consists of the origin and destination vertices, while $\alpha = \emptyset$ in the Minimum Spanning Tree problem (MST). Notice that considering the distance d as a matrix and providing it explicitly in the input avoids (i) providing $(\mathcal{M}, d)$ as part of the input, which may not be straightforward in some cases (e.g., if $(\mathcal{M}, d)$ is a Riemannian manifold), (ii) discussing the complexity of computing the distances which again might involve solving difficult optimization problems.

It has been assumed so far that the position of the vertices are known with precision, which is often not a realistic assumption. In data clustering, uncertainty on the values is common, be it because of a measurement error, or because of a lack of information – in which case the corresponding coordinate is often replaced by the full interval (Masson et al. (2020)). Similar uncertainties arise in network design or facility location because the exact location of the stations and facilities must satisfy technical requirements as well as political considerations as local officials are never happy to let their citizens face the inconvenience of heavy civil engineering. In this paper, we address this issue through the lens of robust optimization, by introducing ROBUST-Π. An instance of ROBUST-Π is given by $I = (\mathbb{G}, \alpha, \mathcal{U}, d)$ where $\mathbb{G}$ and α are as before, d is a distance matrix, $\mathcal{U}_i$ denotes the set of possible positions for node i, which correspond to indexes of rows/columns of the the distance matrix d, and $\mathcal{U} = \times_{i \in \mathbb{V}} \mathcal{U}_i$. Given this instance, the problem is to solve

$$\min_{G \in \mathcal{G}(I)} \max_{u \in \mathcal{U}} \sum_{\{i,j\} \in E[G]} d(u_i, u_j). \qquad \text{(ROBUST-}\Pi\text{)}$$

Thus, the objective of ROBUST-Π is to find $G \in \mathcal{G}(I)$ that minimizes $c(G) = \max_{u \in \mathcal{U}} c(u, G)$. Introducing the notation $\sigma_i = |\mathcal{U}_i|$, we see that ROBUST-$\Pi$ is a generalization of Π as it corresponds to the case where $\sigma_i = 1$ for each i.

Remark 1. *When $(\mathcal{M}, d)$ is the q-dimensional Euclidean space, a natural setting for locational uncertainty would be to model the uncertainty around u_i by a convex set, such as a polytope $\mathcal{P}_i \subset \mathbb{R}^q$, with $\mathcal{P} = \times_{i \in \mathbb{V}} \mathcal{P}_i$. Then, given the set of subgraphs $\mathcal{G}(I)$ of $\mathbb{G}$, one could seek to solve the optimization*

problem

$$\min_{G \in \mathcal{G}(I)} \max_{u \in \mathcal{P}} \sum_{\{i,j\} \in E[G]} d(u_i, u_j), \tag{1}$$

where the adversary maximizes over polytope $\mathcal{P}$ instead of the finite set $\mathcal{U}$ used in (ROBUST-Π). Nevertheless, due to the convexity of d, we have

$$\max_{u \in \mathcal{P}} \sum_{\{i,j\} \in E[G]} d(u_i, u_j) = \max_{u \in \text{ext}(\mathcal{P})} \sum_{\{i,j\} \in E[G]} d(u_i, u_j),$$

where $\text{ext}(\mathcal{P})$ denotes the set of extreme points of $\mathcal{P}$. Therefore, problem (1) can be cast into our setting by letting d be the distance matrix of the set $\bigcup_{i \in \mathbb{V}} \text{ext}(\mathcal{P}_i)$ and defining the sets $\mathcal{U}_i$ accordingly.

Problem ROBUST-Π is closely related to min-max robust combinatorial optimization pioneered in Kouvelis and Yu (2013). In that framework, one searches for the best solution to a combinatorial optimization problem given that the adversary chooses the worst possible cost vector in a given uncertainty set, see the surveys by Aissi et al. (2009); Buchheim and Kurtz (2018); Kasperski and Zieliński (2016). The complexity and approximation ratios available for the resulting problems typically depend on the underlying combinatorial optimization problem (herein represented by Π) and the structure of the uncertainty set. While some algorithms have been proposed to handle arbitrary finite uncertainty sets (e.g. Chassein et al. (2020); Kasperski and Zieliński (2013)), simpler sets typically benefit from stronger results, such as axis-parallel ellipsoids (Baumann et al. (2014); Nikolova (2010)) or budget uncertainty sets (Bertsimas and Sim (2003, 2004)). The latter results have been extended to more general polytopes in Omer et al. (2024) and to problems featuring integer decision variables in Goetzmann et al. (2011). Specific combinatorial optimization problems, such as the shortest path or the spanning tree, have also benefited from stronger results, e.g. Aissi et al. (2005); Yaman et al. (2001).

An important specificity of problem ROBUST-Π is that its cost function is in general non-concave, hardening the inner maximization problem. In fact, we have shown in our companion paper (Bougeret et al. (2023)) that computing the cost function $c(G)$ of a given subgraph $G \in \mathcal{G}(I)$ is in general $\mathcal{NP}$-hard. We have further shown in that paper that problem ROBUST-Π is $\mathcal{NP}$-hard for the shortest path and the minimum spanning tree. That work also proposes exact and approximate solution algorithms based on mixed-integer formulations. Another paper addressing ROBUST-Π is Citovsky et al. (2017), which relies on computational geometry techniques to provide constant-factor approximation algorithms in the special case where Π is the traveling salesman problem, meaning that $\mathcal{G}(I)$ consists of all Hamiltonian cycles of $\mathbb{G}$. The main result of that paper is a polynomial-time approximation scheme for the special case in which each $\mathcal{U}_i$ is a set of disjoint unit disks in the plane. They also propose an approximation algorithm that amounts to solve a deterministic counterpart of ROBUST-Π where the uncertain distances are replaced by the maximum pairwise distances

$$d_{ij}^{max} = \max_{u_i \in \mathcal{U}_i, u_j \in \mathcal{U}_j} d(u_i, u_j),$$

for each $(i, j) \in \mathbb{V}^2, i \neq j$.

In Section 2, we deal with the main purpose of this paper, which is to extend the approximation algorithm based on d^{max} and suggested by Citovsky et al. (2017) to general sets $\mathcal{G}(I)$ and distances matrices more general than those induced by embeddings into Euclidean spaces. First, Theorem 1 transfers any

approximation ratio known for problem $\Pi \in \mathcal{S}$ to ROBUST-Π. We prove that, in general, the transfer involves a multiplicative constant equal to 9 (Theorem 4). Then, we dig into smaller multiplicative constants, assuming that $\mathcal{G}(I)$ and/or d satisfy additional assumptions. Regarding $\mathcal{G}(I)$, we obtain smaller constants for special families of graphs such as cycles, stars, trees, and graphs that can be composed as disjoint unions of these graphs. As a special case of our results, we find the constant of 3 for Hamiltonian cycles previously provided by Citovsky et al. (2017). Concerning the structure of d, we show that distances that satisfy Ptolemy's inequality (see Apostol (1967)) benefit from stronger results.

We complement these results by Section 3 where we provide a dynamic programming algorithm for the special case where Π is the Shortest Path problem, so $\mathcal{G}(I)$ consists of all $s - t$ paths. The algorithm is then turned into a fully-polynomial time approximation scheme by rounding the input appropriately.

Additional notations and definitions Given a simple undirected graph G, $V[G]$ denotes its set of vertices. When clear from context we use notations $n = |V[G]|$ and $m = |E[G]|$, where $|S|$ denotes the cardinality of any finite set S. For any $i \in V[G]$, we denote by $N(i) = \{j \in V[G] \mid \{i, j\} \in E[G]\}$ the neighborhood of i. We say that a graph G is a clique if for any two disjoint vertices $i, j, \{i, j\} \in E$, and we say that G is a star if G is a complete bipartite graph $K_{1,k}$ for some $k \geq 1$. For any positive integer k, we denote $[k] = \{1, \ldots, k\}$. The diameter of a subet S of the indices of d is given by $\max_{u,v \in S} d(u, v)$. Remember also that given a problem ROBUST-Π, $\mathbb{G}$ denotes the graph in the input I, $\mathcal{G}(I)$ is the set of all subgraphs (not necessarily induced) G of $\mathbb{G}$ that satisfy the constraints specific to Π for input I.

2   Approximation of the general robust problem

2.1   Reduction to a deterministic problem by using worst-case distances

---

Algorithm 1: Solving a deterministic counterpart based on some representative location u'

Given an instance $I = (\mathbb{G}, \alpha, \mathcal{U}, d)$ of ROBUST-Π
Select $u' \in \mathcal{U}$ to define instance $I' = (\mathbb{G}, \alpha, u', d')$ of Π, where d' is the submatrix of d
  corresponding to u'
Compute G using an approximation algorithm for I'
return G

A natural approach to ROBUST-Π would be to choose a relevant vector $u' \in \mathcal{U}$, reduce to the corresponding deterministic problem Π given by $\min_{G \in \mathcal{G}(I)} \sum_{\{i,j\} \in E[G]} d\left(u'_i, u'_j\right)$, and use any known approximation algorithm for Π. This approach is formalized by Algorithm 1.

Unfortunately, choosing such a representative u' is not easy. For instance, a natural choice might be to consider the geometric median of each set, i.e., $u' = u^{gm}$ where $u_i^{gm} \in \arg\min_{u_1 \in \mathcal{U}_i} \sum_{u_2 \in \mathcal{U}_i} d\left(u_1, u_2\right)$. Although the choice of geometric median may appear natural at first glance, the cost of the solution G^{gm} returned by Algorithm 1 for $u' = u^{gm}$ may actually be arbitrarily larger than the optimal solution cost.

Observation 1. *Let $\Pi \in \mathcal{S}$ such that any $G \in \mathcal{G}(I)$ is a single edge for any instance I of Π. Let e^{gm} be the solution returned by Algorithm 1 for $u' = u^{gm}$. The ratio $c(e^{gm})/\text{OPT}$ is unbounded.*

Proof: For any $\epsilon > 0$ small enough, consider $\mathbb{V} = \{1, 2, 3\}$ and $\mathbb{E} = \{\{1, 2\}, \{2, 3\}\}$. We consider an embedding into the 1-dimensional Euclidean space with $\mathcal{U}_1 = \{\epsilon\}$, $\mathcal{U}_2 = \{0\}$, and $\mathcal{U}_3 = \{-1, 0, 1\}$, and

defining d as the resulting distance matrix. We have that $u_1^{gm} = \epsilon$ and $u_2^{gm} = u_3^{gm} = 0$ so Algorithm 1 picks edge $\{2,3\}$ having a cost of $c(\{2,3\}) = 1$. In contrast edge $\{1,2\}$ has a cost of ϵ, yielding a ratio of $1/\epsilon$. □

We proceed by using a different approach for reducing to the deterministic problem Π. Given an instance $I = (\mathbb{G}, \alpha, \mathcal{U}, d)$ of ROBUST-Π, we construct an instance $I' = (\mathbb{G}, \alpha, v, d^{max})$ of Π by defining $d_{ij}^{max} = \max_{u_i \in \mathcal{U}_i, u_j \in \mathcal{U}_j} d(u_i, u_j)$ and $v = (1, \ldots, |\mathbb{V}|)$. Notice that d^{max} is indeed a distance matrix as $d_{ij}^{max} = 0$ iff $i = j$ and it satisfies the triangular inequality. This leads to Algorithm 2.

Algorithm 2: Solving a deterministic counterpart based on d^{max} distances

---

Given an instance $I = (\mathbb{G}, \alpha, \mathcal{U}, d)$ of ROBUST-Π
Construct d^{max} to define instance $I' = (\mathbb{G}, \alpha, v, d^{max})$ of Π
Compute G using an approximation algorithm for I'
return G

Theorem 1. *Let $\Pi \in \mathcal{S}$ and assume that:*

- *Π has a ρ_1-polynomial time approximation algorithm, and*

- *there exists $\rho_2 \geq 1$ such that for each input $I = (\mathbb{G}, \alpha, \mathcal{U}, d)$ of ROBUST-Π, $c_{d^{max}}(G) \leq \rho_2 c_d(G)$ for any $G \in \mathcal{G}(I)$.*

Then, Algorithm 2 can be used to derive a polynomial $\rho_1 \rho_2$-approximation for ROBUST-Π.

Proof: Let G^* be an optimal solution of instance $(\mathbb{G}, \alpha, \mathcal{U}, d)$ of ROBUST-Π, and G^{max} be an optimal solution to instance $I' = (\mathbb{G}, \alpha, v, d^{max})$ of Π. Furthermore, following the first assumption, there is a ρ_1-approximation algorithm for Π, which we use to construct a solution G to I' such that $c_{d^{max}}(G) \leq \rho_1 c_{d^{max}}(G^{max})$. We have

$$c_d(G) \leq c_{d^{max}}(G) \leq \rho_1 c_{d^{max}}(G^{max}) \leq \rho_1 c_{d^{max}}(G^*) \leq \rho_1 \rho_2 c_d(G^*),$$

where the last inequality follows from the second assumption of the theorem. □

Let us recall Ptolemy's inequality (see e.g. Apostol (1967)). A distance is *Ptolemaic* if for any four indexes A, B, C, D of d,

$$d(A, C) \cdot d(B, D) \leq d(A, B) \cdot d(C, D) + d(B, C) \cdot d(A, D). \tag{2}$$

In Section 2.2 and Section 2.3 we prove the existence of constant upper bounds on the ratio $c_{d^{max}}(G)/c(G)$ for different families $\mathcal{G}(I)$ and distances. A summary of our results is given in Table 1. In particular, for any graph G, Theorem 3 states that $c_{d^{max}}(G) \leq 4c(G)$ for any Ptolemaic distance, and Theorem 4 states that $c_{d^{max}}(G) \leq 9c(G)$ for any distance. These imply that, up to a constant factor, ROBUST-Π is not harder to approximate than Π, as formalized below.

Theorem 2. *Let $\Pi \in \mathcal{S}$ be a problem that is ρ_1-approximable in polynomial time and $I = (\mathbb{G}, \alpha, \mathcal{U}, d)$ be an instance of ROBUST-Π. Then, Algorithm 2 yields a polynomial $9\rho_1$-approximation for ROBUST-Π, and even a $4\rho_1$-approximation if d is Ptolemaic.*

Problem (each connected component belongs to graph family $\mathcal{G}(I)$)	deterministic version	$c(G)/c_{d^{max}}(G)$	robust counterpart
$\Pi \in \mathcal{S}$ (arbitrary)	ρ-approx.	9 (Thm 4) 4 (Ptolemaic, Thm 3)	9ρ-approx. 4ρ-approx. (Ptolemaic)
MIN-EWCP (clique)	2-approx (Eremin et al. (2014))	2 (Prop 5)	4-approx.
SP (path)	polynomial	2 (Cor 3)	2 approx $\mathcal{NP}$-hard (Bougeret et al. (2023)) $\mathcal{FPTAS}$ (Thm 6)
MST (tree)	polynomial	6 (Prop 9) 4 (Ptolemaic, Thm 3) $2\sqrt{2}$ (planar Eucl., Prop 10)	6-approx. 4-approx. (Ptolemaic) $2\sqrt{2}$-approx. (planar Eucl.) $\mathcal{NP}$-hard (Bougeret et al. (2023))
MIN-SSF (star)	polynomial (Khoshkhah et al. (2019))	3 (Prop 7) 2 (Ptolemaic, Cor 4)	3-approx 2-approx
TSP (cycle)	$\frac{3}{2}$-approx (Christofides (1976))	2 (Citovsky et al. (2017)) and Cor (3)	3-approx (Citovsky et al. (2017)) $\mathcal{PTAS}$ (planar Eucl.) (Citovsky et al. (2017))

Tab. 1: Overview of our results. When no reference is given, the ratios in the "robust counterpart" column are obtained by Theorem 1. All our results are for finite $\mathcal{U}_i$, except the $2\sqrt{2}$ ratio for trees, which holds for balls in the Euclidean plane, and the $\mathcal{PTAS}$ for TSP holding for disjoint unit balls in the Euclidean plane.

2.2 Bounding the approximation ratio on general graphs

We divide our study of $c_{d^{max}}(G)/c(G)$ for arbitrary graphs G into two cases. First, we consider that the distances matrices satisfy the Ptolemy's inequality recalled in (2). Then, we consider arbitrary distances matrices.

2.2.1 Ptolemaic distances

We consider throughout the section that d is Ptolemaic, which includes, for instance, distances resulting from embeddings into Euclidean spaces. A direct consequence of the definition is given by the following lemma.

Lemma 1. *Let d be a Ptolemaic distance matrix and let A, B and C be such that*

$$d(B,C) \geq \max\{d(A,B), d(A,C)\}.$$

Then, for any other O,

$$d(O,A) \leq d(O,B) + d(O,C). \tag{3}$$

Proof: Using $d(B,C) \geq \max\{d(A,B), d(A,C)\}$ and Ptolemy's inequality, we get

$$d(B,C) \cdot d(O,A) \leq d(B,C) \cdot d(O,B) + d(B,C) \cdot d(O,C),$$

and the result follows. $\square$

Using the above inequality, we can get a constant bound on the approximation ratio by focusing on the extremities of a diameter of each uncertainty set $\mathcal{U}_i, i \in \mathbb{V}$. It results in the following ratio.

Theorem 3. *Let d be a Ptolemaic distance and G be a graph. Then, $c_{d^{max}}(G) \leq 4\, c_d(G)$.*

Proof: For all $i \in \{1, \ldots, n\}$, let $[u_i^1, u_i^2]$ be a diameter of $\mathcal{U}_i$, i.e., $u_i^1 \in \mathcal{U}_i, u_i^2 \in \mathcal{U}_i$ and $d\left(u_i^1, u_i^2\right) = \mathrm{diam}(\mathcal{U}_i)$. Let $\widetilde{\mathcal{U}}_i = \{u_i^1, u_i^2\}$ and $\widetilde{\mathcal{U}}$ be the cross product of the $\widetilde{\mathcal{U}}_i$. Let $\tilde{c}(G) = \max_{u \in \widetilde{\mathcal{U}}} \sum_{\{i,j\} \in E[G]} d(u_i, u_j)$. As $\widetilde{\mathcal{U}} \subseteq \mathcal{U}$, we get $c(G) \geq \tilde{c}(G)$. Let us now prove that $\tilde{c}(G) \geq \frac{c_{d^{max}}(G)}{4}$.

Define the random variable $\tilde{u}_i$ taking any value of $\widetilde{\mathcal{U}}_i$ with equal probability $1/2$. The worst-case length of the graph is not smaller than its expected edge length, i.e.,

$$\tilde{c}(G) = \max_{u \in \widetilde{\mathcal{U}}} \sum_{\{i,j\} \in E[G]} d(u_i, u_j) \geq \mathbb{E}\left[\sum_{\{i,j\} \in E[G]} d(\tilde{u}_i, \tilde{u}_j)\right],$$

where, by linearity of expectation,

$$\mathbb{E}\left[\sum_{\{i,j\} \in E[G]} d(\tilde{u}_i, \tilde{u}_j)\right] = \sum_{\{i,j\} \in E[G]} \mathbb{E}\left[d(\tilde{u}_i, \tilde{u}_j)\right].$$

We then consider some arbitrary edge $\{i, j\} \in E[G]$:

$$\mathbb{E}\left[d(\tilde{u}_i, \tilde{u}_j)\right] = \frac{1}{4}\left(d\left(u_i^1, u_j^1\right) + d\left(u_i^1, u_j^2\right) + d\left(u_i^2, u_j^1\right) + d\left(u_i^2, u_j^2\right)\right).$$

Let $\bar{u}_i \in \mathcal{U}_i$ and $\bar{u}_j \in \mathcal{U}_j$ such that $d(\bar{u}_i, \bar{u}_j) = d_{ij}^{max}$. As $u_j^1 u_j^2$ is a diameter of $\mathcal{U}_j$, we have $d(u_j^1, u_j^2) \geq \max(d(u_j^1, \bar{u}_j), d(u_j^2, \bar{u}_j))$, and we can apply Lemma 1 twice to the triplet $(u_j^1, u_j^2, \bar{u}_j)$ to get

$$\begin{cases} d\left(u_i^1, u_j^1\right) + d\left(u_i^1, u_j^2\right) \geq d\left(u_i^1, \bar{u}_j\right) \\ d\left(u_i^2, u_j^1\right) + d\left(u_i^2, u_j^2\right) \geq d\left(u_i^2, \bar{u}_j\right). \end{cases}$$

One last application of the lemma in $u_i^1 u_i^2 \bar{u}_i$ then yields

$$d\left(u_i^1, \bar{u}_j\right) + d\left(u_i^2, \bar{u}_j\right) \geq d(\bar{u}_i, \bar{u}_j) = d_{ij}^{max}.$$

Summarizing the above, we get to $c(G) \geq \tilde{c}(G) \geq \sum_{\{i,j\} \in E[G]} \mathbb{E}\left[d(\tilde{u}_i, \tilde{u}_j)\right] \geq \frac{1}{4} c_{d^{max}}(G)$. $\qquad \square$

2.2.2 Arbitrary distances

Lemma 1 does not apply to non-Ptolemaic distances, as illustrated in the following example.

Example 1. *Consider a distance matrix d based on the index set $\{A, B, C, O\}$ such that for $X \neq Y$ (see also Figure 1)*

$$d(X, Y) = \begin{cases} 1 & \text{if } \{X, Y\} \subseteq \{A, B, C\} \\ 0.5 & \text{if } \{X, Y\} \in \{\{O, B\}, \{O, C\}\} \\ 1.5 & \text{if } \{X, Y\} = \{O, A\} \end{cases},$$

One readily verifies that $d(O, A) = \frac{3}{2}(d(O, B) + d(O, C))$, which contradicts (3).

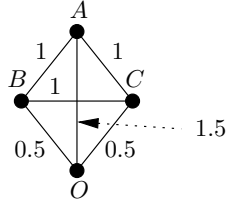In fact, multiplying the right-hand-side of (3) by 3/2, as in Example 1, is enough for any distance matrix.

Fig. 1: Counter example of Lemma 1 for non-Ptolemaic distances.

Lemma 2. *Let (A, B, C) be a triplet such that $d\left(B, C\right) \geq \max\{d\left(A, B\right), d\left(A, C\right)\}$ and $\{O\}$ index another coordinate of d. Then,*

$$d(O, A) \leq \frac{3}{2}(d(O, B) + d(O, C)).$$

Proof: Using $d\left(B, C\right) \geq \max\{d\left(A, B\right), d\left(A, C\right)\}$ and applying the triangle inequality at each step, we get

$$d(O, A) \leq d(O, B) + d\left(A, B\right) \leq d(O, B) + d\left(B, C\right) \leq 2d(O, B) + d(O, C)$$
$$d(O, A) \leq d(O, C) + d\left(A, C\right) \leq d(O, C) + d\left(B, C\right) \leq d(O, B) + 2d(O, C)$$

Adding the above two inequalities provides the result.   □

Using the above result, we can obtain a weaker counterpart of Theorem 3 for non-Ptolemaic distances.

Theorem 4. *For any distance matrix d, $c_{d^{max}}(G) \leq 9\, c_d(G)$.*

Proof: The proof follows exactly the approach followed in the proof of Theorem 3, but we use Lemma 2 instead of Lemma 1 when needed. We thus use the same notations as in the proof of Theorem 3. As $u_j^1 u_j^2$ is a diameter of $\mathcal{U}_j$, we have $d(u_j^1, u_j^2) \geq \max(d(u_j^1, \bar{u}_j), d(u_j^2, \bar{u}_j))$, and we can apply Lemma 2 twice to triplet $(u_j^1, u_j^2, \bar{u}_j)$ to get

$$\begin{cases} d\left(u_i^1, u_j^1\right) + d\left(u_i^1, u_j^2\right) \geq \frac{2}{3} d\left(u_i^1, \bar{u}_j\right) \\ d\left(u_i^2, u_j^1\right) + d\left(u_i^2, u_j^2\right) \geq \frac{2}{3} d\left(u_i^2, \bar{u}_j\right). \end{cases}$$

One last application of Lemma 2 in $u_i^1 u_i^2 \bar{u}_i$ then yields

$$d\left(u_i^1, \bar{u}_j\right) + d\left(u_i^2, \bar{u}_j\right) \geq \frac{2}{3} d\left(\bar{u}_i, \bar{u}_j\right) = \frac{2}{3} d_{ij}^{max}.$$

Summarizing the above, we get to

$$c(G) \geq \tilde{c}(G) \geq \sum_{\{i,j\} \in E[G]} \mathbb{E}\left[d\left(\tilde{u}_i, \tilde{u}_j\right)\right] \geq \frac{1}{4}\frac{4}{9} c_{d^{max}}(G) = \frac{c_{d^{max}}(G)}{9}.$$

                                                                                                □

2.3 Bounding the approximation ratio on specific structures

In what follows, we assume that the structure of the subragph induced by G can be leveraged to obtain stronger bounds than in the previous section. We first describe how graph decomposition can be used to obtain such bounds. We then address the special graphs that have been singled out in our introductory applications, namely: paths, cycles, trees (subway network design), cliques (clustering), and stars (facility location). Unless stated otherwise, we assume throughout the section that d is an arbitrary distance matrix, non-necessarily Ptolemaic.

2.3.1 Building blocks

We study below how the bounds obtained for distinct subgraphs can be combined to obtain a bound on their union.

Proposition 1. *Let $\mathbb{G}$ be a graph, G_t be a subgraph of $\mathbb{G}$, and $\rho_t \geq 1$ such that $c_{d^{max}}(G_t) \leq \rho_t\, c(G_t)$ for each $t = 1, \ldots, T$. Then:*

- $c_{d^{max}}\left(\cup_{t=1}^T G_t\right) \leq T \times \max\limits_{t=1,\ldots,T} \rho_t\, c\left(\cup_{t=1}^T G_t\right)$*, and*

- $c_{d^{max}}(\cup_{t=1}^T G_t) \leq \max\limits_{t=1,\ldots,T} \rho_t\, c(\cup_{t=1}^T G_t)$ *if, in addition, $V(G_t) \cap V(G_{t'}) = \emptyset$ for each $t \neq t' \in [T]$.*

Proof: Let $G = \cup_{t=1}^T G_t$. In the first case we have $T \cdot c(G) \geq \sum\limits_{t=1}^{T} c(G_t) \geq \sum\limits_{t=1}^{T} \frac{1}{\rho_t} c_{d^{max}}(G_t) \geq$

$\dfrac{c_{d^{max}}(G)}{\max\limits_{t=1,\ldots,T} \rho_t}$.

In the second case we have $c(G) = \sum\limits_{t=1}^{T} c(G_t) \geq \sum\limits_{t=1}^{T} \frac{1}{\rho_t} c_{d^{max}}(G_t) \geq \dfrac{c_{d^{max}}(G)}{\max\limits_{t=1,\ldots,T} \rho_t}$. $\qquad\square$

The above results are particularly useful when combining sets G_t having low values of ρ_t. The simplest example of such a set is a single edge.

Observation 2. *For any $e \in E[G]$, $c_{d^{max}}(e) = c(e)$.*

From the above observation and Proposition 1, we obtain immediately that matchings also satisfy the equality.

Corollary 1. *For any matching G, $c_{d^{max}}(G) = c(G)$.*

Matchings can be further combined to obtain general bounds that depend on the characteristics of G. In the remainder, $\chi(G)$ denotes the edge chromatic number of G and $\Delta(G)$ denotes its maximum degree.

Corollary 2. $c_{d^{max}}(G) \leq \chi(G)\, c(G)$.

Proof: By definition of $\chi(G)$, there exists a partition of $E[G]$ into $E_1, \ldots, E_{\chi(G)}$ such that for any $t \in [\chi(G)]$, E_t is a matching. For any $t \in [\chi(G)]$, we define G_t such that G_t only contains edges of E_t (and no other vertices of edges). We have $G = \bigcup_{t=1}^{\chi(G)} G_t$, and by previous corollary on matching, $c_{d^{max}}(G_t) \leq c(G_t)$ for ant t. Thus, by Proposition 1, we get the desired inequality. $\qquad\square$

2.3.2 Graphs with small maximum degree

Recall that Vizing's theorem states that $\chi(G) \leq \Delta(G) + 1$. Combining this with Corollary 2 implies that $c_{d^{max}}(G) \leq (\Delta(G) + 1)c(G)$. Actually, the bound can be decreased to $\Delta(G)$, as stated below.

Proposition 2. *For any graph G, $c_{d^{max}}(G) \leq \Delta(G)\, c(G)$.*

Proof: We follow the proof of Theorem 3 (and re-use same notations), but using different sets $\widetilde{\mathcal{U}}_i$. For any edge $\{i, j\}$ of G, define $u_i^j \in \mathcal{U}_i$ and $u_j^i \in \mathcal{U}_j$ such that $d\left(u_j^i, u_i^j\right) = d_{ij}^{max}$. For any vertex i, let $\widetilde{\mathcal{U}}_i = \{u_i^\ell, \ell \in N(i)\}$, and $\Delta_i = |N(i)|$ be the degree of i. Let $\tilde{c}(G) = \max_{u \in \widetilde{\mathcal{U}}} \sum_{\{i,j\} \in E[G]} d(u_i, u_j)$. As $\widetilde{\mathcal{U}} \subseteq \mathcal{U}$, we know that $c(G) \geq \tilde{c}(G)$. Let us now prove that $\tilde{c}(G) \geq \frac{c_{d^{max}}(G)}{\Delta(G)}$.

For any vertex i, consider random variable $\tilde{u}_i$ taking any value of $\widetilde{\mathcal{U}}_i$ with equal probability. As in the proof of Theorem 3, it is enough to lower bound the following quantity for an arbitrary edge $\{i, j\} \in E[G]$:

$$\mathbb{E}\left[d\left(\tilde{u}_i, \tilde{u}_j\right)\right] = \frac{1}{\Delta_i \Delta_j} \left(\sum_{\ell_1 \in N(i), \ell_2 \in N(j)} d\left(u_i^{\ell_1}, u_j^{\ell_2}\right) \right).$$

Suppose without loss of generality that $\Delta_j \leq \Delta_i$ and let $X_i = N(i) \setminus \{j\}$ and $X_j = N(j) \setminus \{i\}$ (X_i or X_j may be empty). Let us define an arbitrary injective mapping $\phi : X_j \to X_i$, so that

$$\sum_{\ell_1 \in X_i} \sum_{\ell_2 \in X_j} d\left(u_i^{\ell_1}, u_j^{\ell_2}\right) \geq \sum_{\ell_1 \in X_j} \sum_{\ell_2 \in X_j} d\left(u_i^{\phi(\ell_1)}, u_j^{\ell_2}\right) \geq \sum_{\ell_2 \in X_j} d\left(u_i^{\phi(\ell_2)}, u_j^{\ell_2}\right). \tag{4}$$

Using (4), we have

$$
\begin{aligned}
\sum_{\ell_1 \in N(i)\ell_2 \in N(j)} d\left(u_i^{\ell_1}, u_j^{\ell_2}\right) &\geq d_{ij}^{max} + \sum_{\ell \in X_j} d\left(u_i^j, u_j^\ell\right) + \sum_{\ell_1 \in X_i, \ell_2 \in X_j} d\left(u_i^{\ell_1}, u_j^{\ell_2}\right) + \sum_{\ell \in X_i} d\left(u_i^\ell, u_j^i\right) \\
&\geq d_{ij}^{max} + \sum_{\ell \in X_j} d\left(u_i^j, u_j^\ell\right) + \sum_{\ell \in X_j} d\left(u_i^{\phi(\ell)}, u_j^\ell\right) + \sum_{\ell \in X_i} d\left(u_i^\ell, u_j^i\right) \\
&\geq d_{ij}^{max} + \sum_{\ell \in X_j} d\left(u_i^j, u_j^\ell\right) + \sum_{\ell \in X_j} d\left(u_i^{\phi(\ell)}, u_j^\ell\right) + \sum_{\ell \in X_j} d\left(u_i^{\phi(\ell)}, u_j^i\right) \\
&= d_{ij}^{max} + \sum_{\ell \in X_j} \left(d\left(u_i^j, u_j^\ell\right) + d\left(u_i^{\phi(\ell)}, u_j^\ell\right) + d\left(u_i^{\phi(\ell)}, u_j^i\right) \right) \\
&\geq d_{ij}^{max} + \sum_{\ell \in X_j} d_{ij}^{max} = \Delta_j d_{ij}^{max}.
\end{aligned}
$$

We obtain

$$\mathbb{E}\left[d\left(\tilde{u}_i, \tilde{u}_j\right)\right] \geq \frac{1}{\Delta_i \Delta_j} \Delta_j d_{ij}^{max} = \frac{1}{\Delta_i} d_{ij}^{max} \geq \frac{1}{\Delta(G)} d_{ij}^{max}.$$

$\square$

Proposition 2 immediately implies the following Corollary, which had an ad-hoc proof for cycles in Citovsky et al. (2017).

Corollary 3. *Let G be a path or a cycle, then $c_{d^{max}}(G) \leq 2\, c(G)$.*

We verify in the two propositions below that the above bound is tight.

Proposition 3. *For any path G of length at least three, there exists an uncertainty set $\mathcal{U}$ such that $c_{d^{max}}(G) = 2\, c(G)$.*

Proof: We consider a path G composed of $n \geq 3$ vertices, where $E[G] = \{\{i, i+1\} : i \in \{1, \ldots, n-1\}$. The vertices are located on a one-dimensional line where $\mathcal{U}_1 = \{0\}, \mathcal{U}_2 = \{0, 1\}$ and $\mathcal{U}_3 = \cdots = \mathcal{U}_n = \{1\}$. We have $d_{12}^{max} = d_{23}^{max} = 1$ and $d_{i,i+1}^{max} = 0, \forall i = 3, \ldots, n-1$, so $c_{d^{max}}(G) = 2$. There are only two feasible solutions depending on whether $u_2 = 1$ or $u_2 = 0$, and they have the same cost $c(G) = 1$. $\square$

Proposition 4. *For any cycle G of length at least four, there exists an uncertainty set $\mathcal{U}$ such that $c_{d^{max}}(G) = 2\, c(G)$.*

Proof: We consider a cycle G composed of $n \geq 4$ vertices, where $E[G] = \{\{i, i + 1\} : i \in \{1, \ldots, n - 1\} \cup \{\{n, 1\}\}$. The vertices are located on a one-dimensional line where $\mathcal{U}_1 = \{0\}, \mathcal{U}_2 = \{0, 1\}, \mathcal{U}_3 = \{1\}, \mathcal{U}_4 = \{0, 1\}$ and $\mathcal{U}_5 = \cdots = \mathcal{U}_n = \{0\}$ if $n \geq 5$. We verify that $c_{d^{max}}(G) = 4$, and there are four feasible solutions depending on whether $u_2 = 1$ or $u_2 = 0$ and $u_4 = 1$ or $u_4 = 0$. These four solutions all have the same cost $c(G) = 2$. $\square$

The above proposition considers cycles that contain at least four vertices, so one can wonder what happens in the case of smaller cycles. We show next that for cycles that contain only 3 vertices, the bound can be reduced to $3/2$.

Remark 2. *Consider the 3-cycle G. Applying the triangle inequality three times yields:*

$$c(G) = \max_{u \in \mathcal{U}} \left(d(u_1, u_2) + d(u_2, u_3) + d(u_1, u_3) \right) \geq \max_{u \in \mathcal{U}} \left(2 \max\{d(u_1, u_2), d(u_2, u_3), d(u_1, u_3)\} \right)$$

$$\geq 2 \max\{d_{12}^{max}, d_{23}^{max}, d_{13}^{max}\}$$

$$\geq \frac{2}{3} c_{d^{max}}(G),$$

so the maximum worst-case factor is bounded by 3/2. This bound is tight. To see this, one can look at the case exhibited in the proof of Proposition 3 ($\mathcal{U}_1 = \{0\}$, $\mathcal{U}_2 = \{0, 1\}$ and $\mathcal{U}_3 = \{1\}$). For the 3-cycle, $c_{d^{max}}(G) = 3$ and $c(G) = 2$.

2.3.3 Cliques

We now turn to the special case where $\mathcal{G}(I)$ contains only cliques. One specificity of a clique G is that for any matching M of size $\lfloor \frac{n}{2} \rfloor$, every edge of G belongs to a triangle including one edge of M. Applying the triangle inequality repeatedly for a well chosen matching provides the following ratio.

Proposition 5. *Let G be a clique. Then, $c_{d^{max}}(G) \leq 2\, c(G)$.*

Proof: Recall that n denotes $|V[G]|$. It is folklore that $\chi(G) = n - 1$ when n is even, and $\chi(G) = n$ when n is odd, leading to $\chi(G) \leq n$ for any n. This implies that $E[G]$ can be partitioned into n matchings M_i, each of size $\lfloor n/2 \rfloor$, and thus that $c_{d^{max}}(G) = \sum_{i=1}^{n} c_{d^{max}}(M_i)$. Therefore, there is a matching of G, denoted M^*, such that

$$c_{d^{max}}(M^*) \geq \frac{1}{n} c_{d^{max}}(G). \tag{5}$$

Next, we define u^* as any element from $\arg\max\limits_{u \in \mathcal{U}} \sum\limits_{\{i,j\} \in M^*} d(u_i, u_j)$, and we use the shorter notation $d^*_{ij} = d(u^*_i, u^*_j)$, and $d^*_e = d^*_{ij}$ for any edge $e = \{i, j\}$. Observe that because M^* is a matching

$$d^*_e = d^{max}_e \tag{6}$$

for each $e \in M^*$. For any $E' \subseteq E[G]$, let $d^*(E') = \sum_{e \in E'} d^*_e$. Observe that $c(G) \geq d^*(G)$. Our objective is to prove that $d^*(G) \geq \frac{1}{2} c_{dmax}(G)$.

Assume without loss of generality that $M^* = \{\{2i-1, 2i\} \mid i \in [\lfloor \frac{n}{2} \rfloor]\}$, so n is the only vertex not belonging to any edge of M^* when n is odd. For any $i \in [\lfloor \frac{n}{2} \rfloor]$, let $X(2i-1, 2i) = \{\{2i-1, l\} \cup \{2i, l\} \mid l \in V[G] \setminus \{2i-1, 2i\}\}$. As a consequence, $X(2i-1, 2i) \cup \{2i-1, 2i\}$ gathers all the edges of the triangles of G including $\{2i-1, 2i\}$. Observe that the triangle inequality yields

$$d^*(X(2i-1, 2i)) = \sum_{l \in V[G] \setminus \{2i-1, 2i\}} (d^*_{(2i-1)l} + d^*_{(2i)l}) \geq (n-2) d^*_{(2i-1)(2i)}. \tag{7}$$

Summing up (7) for all $i \in [\lfloor \frac{n}{2} \rfloor]$, we obtain

$$\sum_{i \in [\lfloor \frac{n}{2} \rfloor]} d^*(X(2i-1, 2i)) \geq (n-2) d^*(M^*) = (n-2) c_{dmax}(M^*),$$

where the last equality follows from (6). What is more, any edge $e \in E[G] \setminus M^*$ belongs to at most two sets $X(2i-1, 2i)$, so that

$$2d^*(E[G] \setminus M^*) \geq \sum_{i \in [\lfloor \frac{n}{2} \rfloor]} d^*(X(2i-1, 2i)).$$

We obtain

$$2d^*(G) = 2\left[d^*((E[G] \setminus M^*)) + d^*(M^*)\right] \geq (n-2) c_{dmax}(M^*) + 2c_{dmax}(M^*) = n c_{dmax}(M^*).$$

From (5), we finally get $c(G) \geq d^*(G) \geq \frac{n}{2} c_{dmax}(M^*) \geq \frac{1}{2} c_{dmax}(G)$. □

We show below that the above bound is asymptotically tight, even for very simple distances.

Proposition 6. *If G is a k-clique, there exists an uncertainty set $\mathcal{U}$ such that $c_{dmax}(G) = \frac{2(k-1)}{k} c(G)$ if k is odd and $c_{dmax}(G) = \frac{2k}{k+1} c(G)$ if k is even.*

Proof: Let us define $\mathcal{U}_i = \{0, 1\}$ for any $i \in V[G]$, and the distance d by $d(x, y) = |x - y|$. Hence, $c_{dmax}(G) = m$, where $m = |E[G]| = \frac{n(n-1)}{2}$.

On the other hand, we see that computing $c(G)$ is equivalent to partitioning V into $\{V_1, V_2\}$, and defining $u_i = 0$ if $i \in V_1$, and 1 if $i \in V_2$. Hence, $c(G)$ is equal to the optimal cost of a max-cut for G. That cost is equal to $\frac{n^2}{4}$ and $\frac{(n-1)}{2} \frac{(n+1)}{2}$ when n is even and odd, respectively, leading in both cases to the claimed ratio. □

2.3.4 Stars

In what follows, we consider stars whose center is vertex 1 (meaning that for any $i \neq 1$, $|N(i)| = 1$).

Proposition 7. *Let G be a star. Then, assuming that d is Ptolemaic, $c_{d^{max}}(G) \leq 2\, c_d(G)$.*

Proof: Let $\{u_1^1, u_1^2\}$ such that $d\left(u_1^1, u_1^2\right) = \text{diam}(\mathcal{U}_1)$. Let $\bar{u}_1 \in \mathcal{U}_1$ and For all $i \in \{2, \ldots, n\}$, let $\bar{u}_i \in \mathcal{U}_i$ such that there exists $\bar{u}_1 \in \mathcal{U}_1$ with $d(\bar{u}_1, \bar{u}_i) = d_{1i}^{max}$. We then follow the same approach as in the proof of Theorem 3: we set $\overline{\mathcal{U}}_1 = \{u_1^1, u_1^2\}$, $\overline{\mathcal{U}}_i = \{\bar{u}_i\}, \forall i > 1$, and for $i \in V$, we consider the random variables $\tilde{u}_i$ uniformly distributed on $\mathcal{U}_i$. Since $\{u_1^1, u_1^2\}$ is a diameter of $\mathcal{U}_1$, we can apply Lemma 1 to get

$$d(u_1^1, \bar{u}_i) + d(u_1^2, \bar{u}_i) \geq d(\bar{u}_1, \bar{u}_i) = d_{1i}^{max}.$$

This implies $\mathbb{E}\left[d\left(\tilde{u}_1, \tilde{u}_i\right)\right] \geq \frac{1}{2} d_{1i}^{max}$ for any $i > 1$, and thus the claimed ratio. $\qquad\square$

Corollary 4. *Let G be a star. Then, $c_{d^{max}}(G) \leq 3\, c(G)$.*

Proof: The result is obtained with the exact same proof as Proposition 7 where we apply Lemma 2 instead of Lemma 1. $\qquad\square$

We show below that the bound from Corollary 4 is asymptotically tight.

Proposition 8. *Let G be a star on n vertices. There is an uncertainty set $\mathcal{U}$ such that $c_{d^{max}}(G) = \frac{3(n-1)}{n+1}\, c(G)$.*

Proof: We consider the uncertainty set $\mathcal{U} = \bigtimes_{i=1}^{n} \mathcal{U}_i$ where $\mathcal{U}_1 = \{u_1^2, \ldots, u_1^n\}$ and $\mathcal{U}_i = \{u_i\}, i = 2, \ldots, n$, such that:

- for all $(i,j) \in \{2, \ldots, n\}^2, i \neq j, d(u_1^i, u_1^j) = 2/3$,

- for all $(i,j) \in \{2, \ldots, n\}^2, i \neq j, d(u_i, u_j) = 2/3$,

- for all $i = 2, \ldots, n, d(u_1^i, u_i) = 1$ and $\forall j \neq i, d(u_1^j, u_i) = 1/3$.

The triangle inequality is verified, so d is a distance.

By symmetry of the star graph and of the uncertainty set, every solution $u \in \mathcal{U}$ is optimal and has the same value $1 + \frac{1}{3}(n-2)$. Moreover, the maximum distance between $\mathcal{U}_1$ and $\mathcal{U}_i$ is equal to 1 for all $i \in \{2, \ldots, n\}$, so $c_{d^{max}}(G) = n - 1$. As a result, $c_{d^{max}}(G) = \frac{3(n-1)}{n+1} c(G)$. $\qquad\square$

2.3.5 Trees

We conclude our study of specific structures with trees. Our first result combines the bounds obtained for stars in the previous section with the composition results presented in Section 2.3.1 to improve the ratio of 9 obtained in Theorem 4 for general graphs and distances.

Proposition 9. *Let G be a tree. Then $c_{d^{max}}(G) \leq 6\, c(G)$.*

Proof: Observe first that we can partition $E[G]$ into E_1 and E_2 such that each E_i induces a star forest (a graph where any connected component is a star). Indeed, to obtain such a partition we root the tree at vertex 1, and define S^i as the star whose central vertex is i and whose leaves are the children vertices of i in G. Then, we define E_1 (resp. E_2) as the union of $E(S^i)$ for vertices i which are an odd (resp. even) distance from vertex 1. By Proposition 1 and Corollary 4, we get the claimed ratio. $\square$

Next, we show how the bound can be further tightened when considering distances resulting from Euclidean embeddings and spherical uncertainty sets.

Proposition 10. *Assume d is the Euclidean distance, and that for all $i \in V$, $\mathcal{U}_i$ is a closed ball. Then, for any tree G, $c_{d^{max}}(G) \leq 2\sqrt{2}\, c(G)$.*

Proof: We assume without loss of generality that G is rooted at vertex 1. The proof is made by induction on the height of the tree. For this, we consider the following induction statement:

> $P(h)$: *If G has height h, there are two solutions $u^1, u^2 \in \mathcal{U}$ such that*
>
> - $c_{d^{max}}(G) \leq 2\sqrt{2} \sum_{\{i,j\} \in E[G]} d(u_i^1, u_j^1) = \sum_{\{i,j\} \in E[G]} d(u_i^2, u_j^2),$
> - $u_i^1 = u_i^2$ *for each vertex i with level $l < h$,*
> - $[u_i^1, u_i^2]$ *is a diameter of $\mathcal{U}_i$ for each vertex i with level h.*

If $h = 0$, G has only one vertex which is the root of the tree, so the induction statement is trivially satisfied with any diameter $[u^1, u^2]$ of $\mathcal{U}_1$.

Assume that $P(h)$ is true for some $h \geq 1$, and let G be a rooted tree with height $h + 1$. Without loss of generality, we assume that the vertices of $V[G]$ are sorted by increasing level and let n_{h-1}, n_h and n_{h+1} be such that the vertices with level h are $V_h = \{n_{h-1} + 1, \ldots, n_h\}$ and those with level $h + 1$ are $V_{h+1} = \{n_h + 1, \ldots, n_{h+1}\}$. Let $G_{\leq h} = (V_{\leq h}, E_{\leq h})$ be the subtree of G induced by vertices $\{1, \ldots, n_h\}$. Tree $G_{\leq h}$ has height h so we can apply the induction statement to get two solutions $u^1, u^2 \in \mathcal{U}$ as described in $P(h)$.

Now, for all $i \in V_h$, let $S_i \subset E[G]$ be the set of edges of the star graph whose internal vertex is i and whose leaves are the children vertices of i (which all belong to V_{h+1}). Similarly to what was done in the proof of Proposition 7, for all $\{i, j\} \in S_i$, we can set $u_j^1 \in \arg\max_{u \in \mathcal{U}_j} d(u, u_i^1)$ and $u_j^2 \in \arg\max_{u \in \mathcal{U}_j} d(u, u_i^2)$, which yields $d(u_i^1, u_j^1) + d(u_i^2, u_j^2) \geq d_{ij}^{max}$. We can then assume without loss of generality that $\sum_{\{i,j\} \in S_i} d(u_i^1, u_j^1) \geq \sum_{\{i,j\} \in S_i} d(u_i^2, u_j^2)$, i.e.,

$$c_{d^{max}}(S_i) \leq 2 \sum_{\{i,j\} \in S_i} d(u_i^1, u_j^1). \tag{8}$$

Given that we are considering the Euclidean distance with spherical uncertainty sets, the above implies that segment $[u_i^1, u_j^1]$ goes through the center o_j of $\mathcal{U}_j$ (direct application of the triangle inequality). Then, let $[\bar{u}_j^1, \bar{u}_j^2]$ be the diameter of $\mathcal{U}_j$ that is orthogonal to $[u_i^1, u_j^1]$; it exists because $\mathcal{U}_j$ is spherical. Denoting the radius of $\mathcal{U}_j$ by r_j, we then compute

$$\begin{cases} d(u_i^1, u_j^1)^2 = d(u_i^1, o_j)^2 + r_j^2 + 2r_j d(u_i^1, o_j) \leq 2(d(u_i^1, o_j)^2 + r_j^2) \\ d(u_i^1, \bar{u}_j^1)^2 = d(u_i^1, \bar{u}_j^2)^2 = d(u_i^1, o_j)^2 + r_j^2 \end{cases}$$

As a consequence, $d(u_i^1, \bar{u}_j^1) = d(u_i^1, \bar{u}_j^2) \geq 1/\sqrt{2}d(u_i^1, u_j^1)$. Using (8), we then get

$$c_{d^{max}}(S_i) \leq 2\sqrt{2} \sum_{\{i,j\} \in S_i} d(u_i^1, \bar{u}_j^1) = 2\sqrt{2} \sum_{\{i,j\} \in S_i} d(u_i^1, \bar{u}_j^2).$$

The same applies to all i, so we build two solutions $\tilde{u}^1, \tilde{u}^2 \in \mathcal{U}$ such that

- $\tilde{u}_i^1 = \tilde{u}_i^2 = u_i^1$ for all $i \in V_{\leq h}$,

- $[\tilde{u}_i^1, \tilde{u}_i^2] = [\bar{u}_i^1, \bar{u}_i^2]$ is a diameter of $\mathcal{U}_i$ for all $i \in V_{h+1}$.

To conclude, we observe that $E[G] = E_{\leq h} \cup \left(\bigcup_{i \in V_h} S_i \right)$, so

$$\begin{aligned}
\sum_{\{i,j\} \in E[G]} d(\tilde{u}_i^1, \tilde{u}_j^1) &= \sum_{\{i,j\} \in E_{\leq h}} d(\tilde{u}_i^1, \tilde{u}_j^1) + \sum_{i \in V_h} \sum_{\{i,j\} \in S_i} d(\tilde{u}_i^1, \tilde{u}_j^1) \\
&= \sum_{\{i,j\} \in E_{\leq h}} d(u_i^1, u_j^1) + \sum_{i \in V_h} \sum_{\{i,j\} \in S_i} d(u_i^1, \bar{u}_j^1) \\
&\geq \frac{1}{2\sqrt{2}} c_{d^{max}}(G_{\leq h}) + \frac{1}{2\sqrt{2}} \sum_{i \in V_h} c_{d^{max}}(S_i) \\
&= \frac{1}{2\sqrt{2}} c_{d^{max}}(G).
\end{aligned}$$

$\square$

Remark 3. *Notice that Proposition 10 does not fit into the framework described so far as the sets $\mathcal{U}_i$ are infinite. Nevertheless, a finite input for such problems can be defined as $I^{ball} = (\mathbb{G}, \alpha, c, r, \ell)$, where $\mathbb{G}$ and α are as before, c and r denote the center and radii of the balls, respectively, and ℓ denotes their dimension. Furthermore in such problems, we consider that the distance between any two points in $u, v \in \mathbb{R}^\ell$ can be computed in constant time by calling the function $\|u - v\|_2$.*

3 FPTAS for robust shortest path

Up to now, we have mostly focused on general problems ROBUST-Π and provided constant-factor approximation algorithms. The purpose of this section is to focus on a specific problem, ROBUST-SP, known to be $\mathcal{NP}$-hard (Bougeret et al. (2023)). We do this by providing a dynamic programming algorithm for ROBUST-GEN-SP (a generalization of ROBUST-SP where d is not required to verify the triangle inequality) in Section 3.1. As observed at the end of Section 3.1, this dynamic porgramming algorithm becomes polynomial when some parameters are constant (typically $|\mathcal{U}|$). Moreover, it is also used in Section 3.2 to derive the $\mathcal{FPTAS}$.

3.1 Dynamic programming algorithm

In the ROBUST-GEN-SP problem, the input $I = (\mathbb{G}, s, t, \mathcal{U}, d)$ is the same as in the ROBUST-SP problem, except that d is now a non-negative matrix that is only assumed to be symmetric, e.g., $d(u, v) = d(v, u)$ for any $u, v \in \bigcup_{i \in \mathbb{V}} \mathcal{U}_i$. In particular, the function associated with d may not respect the triangle inequality. This level of generality will be useful in the next section when deriving the $\mathcal{FPTAS}$ for ROBUST-SP.

Next, we introduce further notations that are used to derive the dynamic programming algorithm. Let $\mathcal{G}^{i \to t}$ denote the set of all $i - t$ simple paths in $\mathbb{G}$, and $\mathcal{G}_\kappa^{i \to t}$ those having at most κ edges. Given $i \in \mathbb{V}$ and a path $P \in \mathcal{G}^{i \to t}$, we define the worst-case cost given that $u_i = u_i^\ell$ for $\ell \in [\sigma_i]$ (recall that $\sigma_i = |\mathcal{U}_i|$) as

- $c^\ell(P) = \max\{c(u, P) \mid u \in \mathcal{U}, u_i = u_i^\ell\}$, and

- and $Pr(P) = (c^\ell(P))_{\ell \in [\sigma_i]}$ as the profile of P.

We also introduce for any $\kappa \in [n]$ the set of profiles of all $i - t$ paths with at most κ edges as $\mathcal{P}^{(i,\kappa)} = \{Pr(P) \mid P \in \mathcal{G}_\kappa^{i \to t}\}$. We denote

- $\mathrm{Val}(I) = \{c^\ell(P) \mid i \in \mathbb{V}, P \in \mathcal{G}^{i \to t}, \ell \in [\sigma_i]\}$,

- $n_{val} = |\mathrm{Val}(I)|$, and

- $n_{\mathcal{P}} = |\bigcup_{i \in \mathbb{V}, \kappa \in [n]} \mathcal{P}^{(i,\kappa)}|$ the total number of different profiles.

Our objective is to define an algorithm $A(i, \kappa)$ that, given any $i \in \mathbb{V}$ and $\kappa \in [n]$ computes $(\mathcal{P}^{(i,\kappa)}, Q^{(i,\kappa)})$ such that

- $Q^{(i,\kappa)} \subseteq \mathcal{G}_\kappa^{i \to t}$,

- $|Q^{(i,\kappa)}| = |\mathcal{P}^{(i,\kappa)}|$,

- for any $p \in \mathcal{P}^{(i,\kappa)}$, there exists $P \in Q^{(i,\kappa)}$ such that $Pr(P) = p$.

Informally, $A(i, \kappa)$ computes all profiles associated to (i, κ) as well as a representative path for each one of these profiles. Let us first verify that computing this is enough to solve ROBUST-GEN-SP optimally.

Lemma 3. *Given an input $I = (\mathbb{G}, s, t, \mathcal{U}, d)$ of the* ROBUST-GEN-SP *problem, and given $(\mathcal{P}^{(s,n)}, Q^{(s,n)})$, we can find an optimal solution of I in time polynomial in n and linear in $n_{\mathcal{P}}$.*

Proof: For any $p \in \mathcal{P}^{(s,n)}$, $p = (p_\ell)_{\ell \in [\sigma_s]}$, let $x_p = \max_{\ell \in [\sigma_s]} p_\ell$. We define

$$p^{min} = \underset{p \in \mathcal{P}^{(s,n)}}{\arg \min} \, x_p$$

and output $P^{min} \in Q^{(s,n)}$ such that $Pr(P^{min}) = p^{min}$. Let P^* be an optimal solution and $p^* = Pr(P^*)$. As $p^* \in \mathcal{P}^{(s,n)}$, we have $c(P^{min}) = x_{p^{min}} \leq x_{p^*} = c(P^*)$. $\qquad\qquad \square$

We provide next the dynamic programming recursion for $\mathcal{P}^{(i,\kappa)}$, leaving aside the computation of $Q^{(i,\kappa)}$ to simplify the presentation. Given $i \in \mathbb{V}$, $\kappa \in [n]$, $j \in N(i)$, $P' \in \mathcal{G}_\kappa^{j \to t}$, and $p' = Pr(P')$, we consider the $i - t$ path $P = iP'$ obtained by concatenating i with P'. One readily verifies that $Pr(P) = p(i, \kappa, j, p')$, where $p(i, \kappa, j, p') = (y_\ell)_{\ell \in [\sigma_i]}$, with $y_\ell = \max_{\ell' \in [\sigma_j]} d\left(u_i^\ell, u_j^{\ell'}\right) + p'_{\ell'}$. We obtain that for any $i \neq s$ and $\kappa > 0$

$$\mathcal{P}^{(i,\kappa)} = \{p(i, \kappa, j, p') \mid j \in N(i), p' \in \mathcal{P}^{(j,\kappa-1)}\}, \tag{9}$$

and $\mathcal{P}^{(s,0)} = (0)_{\ell \in [\sigma_s]}$. Recall that $\sigma = \max_{i \in \mathbb{V}} \sigma_i$. We provide in the next lemma the complexity of the resulting dynamic programming algorithm.

Lemma 4. *Let $i \in \mathbb{V}$, $\kappa \in [n]$. Given $\mathcal{P}^{(j,\kappa-1)}$ and $Q^{(j,\kappa-1)}$ for any $j \in N(i)$, we can compute $(\mathcal{P}^{(i,\kappa)}, Q^{(i,\kappa)})$ in time $\mathcal{O}(n\sigma^2 n_\mathcal{P})$ and space $\mathcal{O}(n_\mathcal{P} n(log(n)))$.*

Proof: For complexity issues we assume that $\mathcal{P}^{(j,\kappa-1)}$ and $Q^{(j,\kappa-1)}$ are represented as arrays A_P and A_Q indexed by profiles, where given a profile p, $A_P[p]$ is true iff $p \in \mathcal{P}^{(j,\kappa-1)}$, and $A_Q[p]$ contains a path P such that $Pr(P) = p$ if $p \in \mathcal{P}^{(j,\kappa-1)}$, and $\emptyset$ otherwise. This explains the $\mathcal{O}(n_\mathcal{P} n(log(n)))$ required space for storing Q. As profiles are vectors of length at most σ, we consider that it takes $\mathcal{O}(\sigma)$ to obtain the value stored at index p of array A_P or A_Q.

We compute $\mathcal{P}^{(i,\kappa)}$ following the recursion relation (9), and compute $Q^{(i,\kappa)}$ along the way. More precisely, we start by initializing two arrays A'_P and A'_Q of size $n_\mathcal{P}$. Then, for all $j \in N(i)$ and $p' \in \mathcal{P}^{(j,\kappa-1)}$, we compute $p(i,\kappa,j,p')$ in time $\mathcal{O}(\sigma^2)$. Now, we perform the following operations in $\mathcal{O}(1)$. If $p(i,\kappa,j,p')$ is not already in $\mathcal{P}$, we add it to $\mathcal{P}$, we find a path P' in $Q^{(j,\kappa-1)}$ such that $Pr(P') = p'$, and we add the path iP' to Q. $\qquad\square$

We are now ready to state the main result of this section.

Theorem 5. ROBUST-GEN-SP *can be solved in time $\mathcal{O}(n^3\sigma^2 n_\mathcal{P})$ and space $\mathcal{O}(n_\mathcal{P} n^3 log(n))$.*

Proof: We compute $(\mathcal{P}^{(s,n)}, Q^{(s,n)})$ using a DP algorithm based on (9), and obtain an optimal solution by Lemma 3. The overall complexity is dominated by the DP, whose complexity is in $\mathcal{O}(a \times b)$, where a denotes the number of entries of the associated memoization table, and b is the time-complexity of computing one entry, assuming the other ones are accessible in $\mathcal{O}(1)$. As there is an entry (i, κ) for any $i \in \mathbb{V}$ and $\kappa \in [n]$, we get $a = \mathcal{O}(n^2)$. By Lemma 4, we get $b = \mathcal{O}(n\sigma^2 n_\mathcal{P})$ get the claimed time complexity. The space complexity immediately follows. $\qquad\square$

Let us further elaborate on the value of $n_\mathcal{P}$ that arises in Theorem 5. First of all, we see that $n_\mathcal{P} \leq (n_{val})^\sigma$, leading to the observation below, used in the next section to derive the FPTAS.

Observation 3. ROBUST-GEN-SP *can be solved in time $\mathcal{O}(n^3\sigma^2(n_{val})^\sigma)$*

From a more theoretical viewpoint, notice that the reduction of Bougeret et al. (2023)[Proposition 1] proving the hardness of ROBUST-GEN-SP involves a "large" set $\mathcal{U}$, so a natural question is whether ROBUST-GEN-SP becomes polynomial for "small" sets $\mathcal{U}$, either in terms of diameter or number of elements. It so happens that the two questions can be answered positively. Namely, observe that for any value $c^\ell(P)$, we can find some $n_{v,v'} \in [n]$ for each $(v, v') \in \mathcal{U}^2$ such that $c^\ell(P) = \sum_{(v,v') \in \mathcal{U}^2} n_{v,v'} d(v, v')$. Therefore, n_{val} can be bounded by $n^{\frac{|\mathcal{U}|(|\mathcal{U}|-1)}{2}}$, and ROBUST-GEN-SP can be solved in polynomial time if $|\mathcal{U}|$ is constant. Alternatively, if all distances are integer, meaning d has integer values, then $n_\mathcal{P} \leq n \times \mathrm{diam}(\mathcal{U})$, so ROBUST-GEN-SP can be solved in $\mathcal{O}(n^3\sigma^2(n \times \mathrm{diam}(\mathcal{U}))^\sigma)$ in that case.

3.2 FPTAS

We now consider solving ROBUST-SP approximately. More precisely, we provide an algorithm that, given an input I of ROBUST-SP and $\epsilon > 0$, outputs an $(1 + \epsilon)$-approximated solution in time polynomial both in n and $1/\epsilon$. More precisely, given any $\epsilon > 0$, we want to provide an $(1 + \epsilon)$-approximated solution. Let $I = (\mathbb{G}, s, t, \mathcal{U}, d)$ be an input to ROBUST-SP and A be an upper bound to $\mathrm{OPT}(I)$ and $\epsilon' > 0$. We define a matrix d' by rounding each element of d to the closest value of the form $\epsilon'\ell A$ for some $\ell \in \mathbb{N}$. We obtain an instance $I' = (\mathbb{G}, s, t, \mathcal{U}, d')$ to ROBUST-GEN-SP. Having Observation 3 in mind, a straightforward application of the DP from the previous section to I' would yield too many values n_{val}. Hence, we show

in Section 3.2 how to adapt the DP, and choose A and ϵ' (depending on ϵ) appropriately to obtain the result below.

Theorem 6. ROBUST-SP *admits an $\mathcal{FPTAS}$ for fixed σ: for any ϵ, we can compute a $(1+\epsilon)$-approximated solution in time $\mathcal{O}(n^3\sigma^2(\frac{n^2}{\epsilon})^\sigma)$.*

Proof: Let $I = (\mathbb{G}, s, t, \mathcal{U}, d)$ be an instance of ROBUST-SP. Our objective is to provide a solution of cost at most $(1 + \epsilon)\text{OPT}(I)$. Let $\epsilon' \in \mathbb{R}^+$. Using the 2-approximation obtained combining Theorem 1 and Proposition 2, we start by computing an $s - t$ path P_A of cost $c(P_A) = A$, where $\text{OPT}(I) \leq A \leq 2\text{OPT}(I)$. For any $x, y \in \bigcup_{i \in \mathbb{V}} \mathcal{U}_i$, we define $d'(x, y)$ by rounding up $d(x, y)$ to the closest value of the form $\epsilon'\ell A$ for some $\ell \in \mathbb{N}$. For any path P and $u \in \mathcal{U}$ we denote by $c'(u, P) = \sum_{i,j \in P} d'(u_i, u_j)$, $c'(P) = \max_{u \in \mathcal{U}} c'(u, P)$. Let $I' = (\mathbb{G}, s, t, \mathcal{U}, d')$ be the instance of ROBUST-GEN-SP obtained when using d' instead of d.

Observe that

- the function $d'(u, v) = d'(u, v)$ may not be a distance,

- for any $x, y \in \bigcup_{i \in \mathbb{V}} \mathcal{U}_i$, we have $d(u, v) \leq d'(u, v) \leq d(u, v) + \epsilon'A$

- for any path P, $c(P) \leq c'(P) \leq c(P) + n\epsilon'A$

- $\text{OPT}' \leq \text{OPT} + n\epsilon'A$.

Let $i \in \mathbb{V}$ and P be an $i - t$ path. We say that P is *useless* if it verifies $c'(P) > A(1 + n\epsilon')$; otherwise, P is said to be *good*. According to previous observations, we see that $c'(P_A) \leq A(1 + n\epsilon')$. Thus, for any $i \in \mathbb{V}$ and useless $i - t$ path P, we have $c'(P) > c'(P_A)$. This implies that P cannot be the suffix of an optimal solution to input I' (meaning that there is no optimal solution of I' that first goes from s to i, and then uses P). As a consequence, in the DP algorithm provided in Section 3.1, we can restrict our attention to the profiles of good paths, without loosing optimality in I'. More formally, for any $i \in \mathbb{V}$ and $\kappa \in [n]$, we adapt the previous definition $\mathcal{P}^{(i,\kappa)}$ to

$$\mathcal{P}_{\text{good}}^{(i,\kappa)} = \{Pr(P) \mid P \in \mathcal{G}_\kappa^{i \to t}, P \text{ is good}\},$$

and we now consider that the DP algorithm $A_g(i, \kappa)$ computes $(\mathcal{P}_{\text{good}}^{(i,\kappa)}, Q^{(i,\kappa)})$ instead of $(\mathcal{P}^{(i,\kappa)}, Q^{(i,\kappa)})$. We now compute P^* an optimal solution on instance I' (for cost function c') using Theorem 5 (with DP algorithm A_g), and output P^*.

We have $c(P^*) \leq c'(P^*) = \text{OPT}(I') \leq \text{OPT}(I) + n\epsilon'A \leq \text{OPT}(I)(1 + 2n\epsilon')$. Let us now consider the complexity of computing P^*. As $c'(P) = \max_{\ell \in [\sigma_i]} c'^\ell(P)$, observe that for any good path P we have $c'^\ell(P) \leq A(1 + n\epsilon')$. Moreover, as for any $x, y \in \bigcup_{i \in \mathbb{V}} \mathcal{U}_i$, $d'(u, v)$ is a multiple of $\epsilon'A$, we get that for any good path P, $c'^\ell(P) = \ell\epsilon'A$ for $0 \leq l \leq n + \lceil \frac{1}{\epsilon'} \rceil$. This implies that $n_{val} \leq (n + 2 + \frac{1}{\epsilon'}) = \mathcal{O}(\frac{n}{\epsilon'})$, so Observation 3 leads to the desired complexity of $\mathcal{O}(n^3\sigma^2(\frac{n}{\epsilon'})^\sigma)$ to get a ratio $1 + 2n\epsilon'$. Finally, given a target ratio $1 + \epsilon$, we set $\epsilon' = \frac{\epsilon}{2n}$, obtaining the claimed complexity. $\qquad\qquad\square$

4 Conclusion

In this paper, we proved that for a certain type of problem $\Pi \in \mathcal{S}$, we can transfer the approximability of Π to its robust variant ROBUST-Π (to the price of a constant multiplicative factor), and we provided an $\mathcal{FPTAS}$ for the robust shortest path problem. Natural questions for future research would be to improve some of the multiplicative constants we obtained for specific structures and to generalize the $\mathcal{FPTAS}$ to a larger class of problems.

References

H. Aissi, C. Bazgan, and D. Vanderpooten. Approximation complexity of min-max (regret) versions of shortest path, spanning tree, and knapsack. In G. S. Brodal and S. Leonardi, editors, *Algorithms - ESA 2005, 13th Annual European Symposium*, volume 3669 of *Lecture Notes in Computer Science*, pages 862–873, 2005.

H. Aissi, C. Bazgan, and D. Vanderpooten. Min-max and min-max regret versions of combinatorial optimization problems: A survey. *European Journal of Operational Research*, 197(2):427–438, 2009.

T. M. Apostol. Ptolemy's Inequality and the Chordal Metric. *Mathematics Magazine*, 40(5):233–235, 1967.

F. Baumann, C. Buchheim, and A. Ilyina. Lagrangean decomposition for mean-variance combinatorial optimization. In *ISCO*, pages 62–74, 2014.

D. Bertsimas and M. Sim. Robust discrete optimization and network flows. *Mathematical Programming*, 98(1-3):49–71, 2003.

D. Bertsimas and M. Sim. The price of robustness. *Operations Research*, 52(1):35–53, 2004.

M. Bougeret, J. Omer, and M. Poss. Optimization problems in graphs with locational uncertainty. *INFORMS Journal on Computing*, 35(3):578–592, 2023.

C. Buchheim and J. Kurtz. Robust combinatorial optimization under convex and discrete cost uncertainty. *EURO J. Computational Optimization*, 6(3):211–238, 2018.

A. B. Chassein, M. Goerigk, A. Kasperski, and P. Zieliński. Approximating combinatorial optimization problems with the ordered weighted averaging criterion. *European Journal of Operational Research*, 286(3):828–838, 2020.

N. Christofides. Worst-case analysis of a new heuristic for the traveling salesman problem. In *Symposium on new directions and recent results in algorithms and complexity*, page 441, 1976.

G. Citovsky, T. Mayer, and J. S. B. Mitchell. TSP With Locational Uncertainty: The Adversarial Model. In B. Aronov and M. J. Katz, editors, *33rd International Symposium on Computational Geometry (SoCG 2017)*, volume 77 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 32:1–32:16, 2017.

I. Eremin, E. K. Gimadi, A. Kel'manov, A. Pyatkin, and M. Y. Khachai. 2-approximation algorithm for finding a clique with minimum weight of vertices and edges. *Proceedings of the Steklov Institute of Mathematics*, 284(1):87–95, 2014.

K. Goetzmann, S. Stiller, and C. Telha. Optimization over integers with robustness in cost and few constraints. In *WAOA 2011, Saarbrücken, Germany, September 8-9, 2011, Revised Selected Papers*, pages 89–101, 2011.

G. Gutiérrez-Jarpa, C. Obreque, G. Laporte, and V. Marianov. Rapid transit network design for optimal cost and origin–destination demand capture. *Computers & Operations Research*, 40(12):3000–3009, 2013.

A. Kasperski and P. Zieliński. Bottleneck combinatorial optimization problems with uncertain costs and the OWA criterion. *Operations Research Letters*, 41(6):639–643, 2013.

A. Kasperski and P. Zieliński. Robust discrete optimization under discrete and interval uncertainty: A survey. In *Robustness analysis in decision aiding, optimization, and analytics*, pages 113–143. Springer, 2016.

K. Khoshkhah, M. Khosravian Ghadikolaei, J. Monnot, and D. O. Theis. Complexity and approximability of extended spanning star forest problems in general and complete graphs. *Theoretical Computer Science*, 775:1–15, 2019.

P. Kouvelis and G. Yu. *Robust discrete optimization and its applications*, volume 14. Springer Science & Business Media, 2013.

M.-H. Masson, B. Quost, and S. Destercke. Cautious relational clustering: A thresholding approach. *Expert Systems with Applications*, 139:112837, 2020.

S. Melkote and M. S. Daskin. An integrated model of facility location and transportation network design. *Transportation Research Part A: Policy and Practice*, 35(6):515–538, 2001.

E. Nikolova. Approximation algorithms for reliable stochastic combinatorial optimization. In *APPROX-RANDOM*, pages 338–351, 2010.

J. Omer, M. Poss, and M. Rougier. Combinatorial Robust Optimization with Decision-Dependent Information Discovery and Polyhedral Uncertainty. *Open Journal of Mathematical Optimization*, 5:5, 2024. doi: 10.5802/ojmo.33.

M. Poss. Robust combinatorial optimization with knapsack uncertainty. *Discrete Optimization*, 27:88–102, 2018.

H. Yaman, O. E. Karasan, and M. Ç. Pinar. The robust spanning tree problem with interval data. *Operations Research Letters*, 29(1):31–40, 2001.

J. Zhen, F. J. de Ruiter, E. Roos, and D. den Hertog. Robust optimization for models with uncertain second-order cone and semidefinite programming constraints. *INFORMS Journal on Computing*, 34 (1):196–210, 2021.