BULK-ROBUST ASSIGNMENT PROBLEMS: HARDNESS, APPROXIMABILITY AND ALGORITHMS

Dissertation

zur Erlangung des akademischen Grades eines
Doktors der Naturwissenschaften
(Dr. rer. nat.)

Der Fakultät für Mathematik der
Technischen Universität Dortmund
vorgelegt von

Viktor Bindewald

im Oktober 2017

Dissertation

Bulk-Robust Assignment Problems: Hardness, Approximability and Algorithms

Fakultät für Mathematik
Technische Universität Dortmund

Erstgutachter: JP Dr. Dennis Michaels

Zweitgutachter: Prof. Dr. Volker Kaibel

Tag der mündlichen Prüfung: 18. Dezember 2017

ABSTRACT

This thesis studies robust assignment problems with focus on computational complexity. Assignment problems are well-studied combinatorial optimization problems with numerous practical applications, for instance in production planning.

Classical approaches to optimization expect the input data for a problem to be given precisely. In contrast, real-life optimization problems are modeled using forecasts resulting in uncertain problem parameters. This fact can be taken into account using the framework of robust optimization.

An instance of the classical assignment problem is represented using a bipartite graph accompanied by a cost function. The goal is to find a minimum-cost assignment, i.e., a set of resources (edges or nodes in the graph) defining a maximum matching. Most models for robust assignment problems suggested in the literature capture only uncertainty in the costs, i.e., the task is to find an assignment minimizing the cost in a worst-case scenario. The contribution of this thesis is the introduction and investigation of the Robust Assignment Problem (RAP) which models edge and node failures while the costs are deterministic. A scenario is defined by a set of resources that may fail simultaneously. If a scenario emerges, the corresponding resources are deleted from the graph. RAP seeks to find a set of resources of minimal cost which is robust against all possible incidents, i.e., a set of resources containing an assignment for all scenarios. In production planning for example, lack of materials needed to complete an order can be encoded as an edge failure and production line maintenance corresponds to a node failure.

The main findings of this thesis are hardness of approximation and NP-hardness results for both versions of RAP, even in case of single edge (or node) failures. These results are complemented by approximation algorithms matching the theoretical lower bounds asymptotically. Additionally, we study a new related problem concerning k-robust matchings. A perfect matching in a graph is k-robust if the graph remains perfectly matchable after the deletion of any k matching edges from the graph. We address the following question: How many edges have to be added to a graph to make a fixed perfect matching k-robust? We show that, in general, this problem is as hard as both aforementioned variants of RAP. From an application point of view, this result implies that robustification of an existent infrastructure is not easier than designing a new one from scratch.

ZUSAMMENFASSUNG

Diese Dissertation behandelt robuste Zuordnungsprobleme mit dem Schwerpunkt auf deren komlexitätstheoretischen Eigenschaften. Zuordnungsprobleme sind gut untersuchte kombinatorische Optimierungsprobleme mit vielen praktischen Anwendungen, z. B. in der Produktionsplanung.

Klassische Ansätze der Optimierung gehen davon aus, dass die Inputdaten eines Problems exakt gegeben sind, wohingegen Optimierungsprobleme aus der Praxis mit Hilfe von Voraussagen modelliert werden. Daraus folgen unsichere Problemparameter, woran die Robuste Optimierung ansetzt. Die Unsicherheit wird mit Hilfe einer Szenarienmenge modelliert, die alle möglichen Ausprägungen der Problemparameter beschreibt.

Eine Instanz des klassischen Zordnungsproblems wird mit Hilfe eines Graphen und einer Kostenfunktion beschrieben. Die Aufgabe besteht darin, eine Zuordnung mit minimalen Kosten zu finden. Eine Zuordnung ist eine Teilmenge an Ressourcen (Kanten oder Knoten des Graphen), die ein kardinalitätsmaximales Matching induziert. In der Literatur sind überwiegend robuste Zuordnungsprobleme untersucht, die Unsicherheit in den Kosten behandeln, in diesem Fall besteht die Aufgabe darin, eine Zuordnung mit minimalen Kosten im Worst-Case-Szenario zu finden. Diese Dissertation dient der Einführung und Untersuchung des *Robust Assignment Problem* (RAP) welches Kanten- und Knotenausfälle modelliert; wobei die Kosten determinisitsch sind. Ein Szenario ist durch jene Teilmenge an Ressourcen definiert, welche gleichzeitig ausfallen können. Wenn ein Szenario eintritt, werden die jeweils ausfallenden Ressourcen aus dem Graphen entfernt. In RAP besteht das Ziel darin, eine Menge an Ressourcen mit minimalen Kosten zu finden, die robust gegenüber allen möglichen Ereignissen ist, d. h. eine Ressourcenmenge die für alle Szenarien eine gültige Zuordnung enthält. So kann beispielsweise in der Produktionsplanung der Mangel an Materialien, die für einen Auftrag benötigt werden, als Kantenausfall und die wartungsbedingte Abschaltung einer Produktionslinie als Knotenausfall modelliert werden.

Die Hauptergebnisse dieser Arbeit sind Nichtapproximierbarkeits- und NP-Schwierigkeitsresultate beider RAP-Versionen, die bereits für die Einschränkung zutreffen, dass nur einzelne Kanten oder Knoten ausfallen können. Diese Ergebnisse werden durch Approximationsalgorithmen ergänzt, die die theoretischen Approximationsschranken asymptotisch erreichen. Zusätzlich wird ein neues, verwandtes Optimierungsproblem untersucht, welches sich mit k-robusten Matchings

beschäftigt. Ein perfektes Matching in einem Graphen ist k-robust, wenn der Graph nach dem Löschen von k Matchingkanten weiterhin ein perfektes Matching besitzt. Es wird der Frage nachgegangen, wie viele Kanten zum Graphen hinzugefügt werden müssen, um ein gegebenes Matching k-robust zu machen. Dabei wird gezeigt, dass dieses Problem im Allgemeinen aus komplexitätstheoretischer Sicht genauso schwierig ist, wie die zuvor erwähnten RAP-Varianten. Aus der Anwendungsperspektive bedeutet dieses Resultat, dass die Robustifikation einer bestehender Infrastruktur nicht einfacher ist, als sie von Grund auf neu zu entwerfen.

PARTIAL PUBLICATIONS AND COLLABORATION PARTNERS

The majority of the results in Chapter 3 and 4 were developed together with David Adjiashvili and my supervisor Dennis Michaels and can be found in [ABM16a], [ABM16b] and [ABM17]. Most of the results in Chapter 5 were developed together with Moritz Mühlenthaler and were partially published in [BM17].

[ABM16a] David Adjiashvili, Viktor Bindewald, and Dennis Michaels. "Robust Assignments via Ear Decompositions and Randomized Rounding." *Proceedings of the 43rd International Colloquium on Automata, Languages, and Programming (ICALP)*. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2016, 71:1–71:14. DOI: 10.4230/LIPIcs.ICALP.2016.71.

[ABM16b] David Adjiashvili, Viktor Bindewald, and Dennis Michaels. "Robust Assignments via Ear Decompositions and Randomized Rounding." *arXiv preprint*, 2016. Full version of [ABM16a]. URL: http://arxiv.org/abs/1607.02437.

[ABM17] David Adjiashvili, Viktor Bindewald, and Dennis Michaels. "Robust Assignments with Vulnerable Nodes." *arXiv preprint*, 2017. URL: http://arxiv.org/abs/1703.06074.

[BM17] Viktor Bindewald and Moritz Mühlenthaler. "Robust Bipartite Matching Augmentation." *15th Cologne-Twente Workshop on Graphs and Combinatorial Optimization (CTW)*. Extended abstract. 2017, pp. 11–14.

ACKNOWLEDGMENTS

First of all I like to thank my supervisor Dennis Michaels for his scientific guidance and advice. I owe a lot to all my colleagues I have met during my time at TU Dortmund for their help and numerous fruitful discussions. I am also very grateful to Christoph Buchheim for creating a fantastic working atmosphere in the group. Special thanks go to Sabine Willrich for always helping with administrative issues. Moreover, I owe many thanks to David Adjiashvili and Moritz Mühlenthaler for productive cooperation.

The support of my work by the German Research Foundation (DFG) within the Research Training Group *"Discrete Optimization of Technical Systems under Uncertainty"* is gratefully acknowledged. Parts of this work were carried out while I was visiting the Institute for Operations Research (IFOR) at ETH Zurich. The hospitality is highly appreciated.

Many thanks go to my friends for accompanying and encouraging me during my PhD. Finally, I like to thank my family and Nele for their absolute support.

CONTENTS

LIST OF ALGORITHMS

LIST OF COMPUTATIONAL PROBLEMS

1

INTRODUCTION

In our everyday life we perform numerous tasks which can be classified as combinatorial optimization problems. Typical examples are finding a fast route to work, packing bags or scheduling appointments. In real life, the information describing a particular task is uncertain in nature. For example, the journey time to your workplace heavily depends on the traffic conditions and can be disrupted by single events such as road accidents. However, this fact is often neglected by the classical optimization theory. One possibility to integrate uncertainty into decision making processes is given by the robust optimization framework.

In this thesis we study robust assignment problems. In practice, assignment problems can be encountered in various contexts. An evident example is the allocation of orders to production lines in production planning. But assignment problems can also occur, in a less obvious way, as subproblems of other optimization problems such as the Traveling Salesperson Problem [Chr76] or vehicle routing [BDM12, pp. 65-66]. For more examples of combinatorial problems with assignment-like structure the reader is referred to [Woe07]. The goal of the classical assignment problem (AP) is to pair up objects from two different classes while maximizing the number of pairs or minimizing the costs.

The input data for an assignment problem is usually represented by a bipartite graph $G = (U \dot\cup W, E)$ and a linear cost function c on the edge set E. In general, both the graph and the costs can be affected by uncertainty. Most of the robust assignment problems studied in the literature treat uncertainty in the cost structure only (see the overview in Section 2.6.1). In this thesis, we study assignment problems with edge and node failures, i.e., the graph's structure is subject to uncertainty. We implement the recently introduced bulk robustness concept [ASZ15], where a deterministic optimization problem is accompanied by a finite set $\mathfrak{S}$ of failure scenarios. Each scenario is specified by a subset of resources R, where the choice of R depends on the nominal problem. Here, we consider a variant of AP that seeks to find a minimum-cost assignment covering all nodes in U as the nominal problem and distinguish two natural versions. The goal is either to select a U-perfect matching or a subset of the node set W such that

each node in U can be matched to one of the nodes selected from W. This means that the set of resources is either E or W. After the realization of a scenario $F \subseteq R$, the corresponding resources are removed from the graph. The resulting Robust Assignment Problem reads as

$$\begin{aligned} \min \quad & c(X) \\ \text{s.t.} \quad & \forall\, F \in \mathfrak{S}\colon G[X] - F \text{ contains a } U\text{-perfect matching}, \\ & X \subseteq R. \end{aligned} \tag{RAP}$$

Depending on the type of resources R we differentiate between two variants of RAP. If $R = E$, then we call the resulting problem E-RAP and if $R = V$ we call it V-RAP. In general, solutions to RAP include resources that are part of some failure scenario and are not feasible to the underlying deterministic problem. This means RAP fits into the class of redundancy-based robust optimization problems. We present some potential applications by the end of this section.

In both variants of RAP the goal is to design a fault-tolerant structure. In practice, however, it may be more preferable to make an existent infrastructure robust against incidents rather then redesigning it entirely. The notion of robustness for a fixed matching is formalized as follows. Given a bipartite graph $G = (U \dot\cup W, E)$ and a fixed perfect matching M in G, we require G to remain perfectly matchable after the removal of k arbitrary M-edges from G. A matching with this property is called k-robust. Expecting an adversary to remove k edges from M, we seek to robustify M by adding new edges to the graph G. This motivational question leads to the Robust Matching Augmentation Problem defined as

$$\begin{aligned} \min \quad & c(L) \\ \text{s.t.} \quad & M \text{ is a } k\text{-robust matching in } G + L, \\ & L \subseteq \big\{\{u, w\}\colon u \in U,\ w \in W\big\} \setminus E. \end{aligned} \tag{RMAP}$$

Additionally, we may prescribe that the two matchings, M and its replacement after the edge removal, do not differ too much by specifying a recovery budget.

Before outlining the structure of this thesis we provide motivational application examples for each of the three problems described above.

Continuity of Service (E-RAP). In the service sector stable client-provider relationships are favorable for both, the customers and the businesses. Patients of a hospital for example feel more secure if the nursing staff do not change frequently. The employers in the consulting industries benefit from increased turnover due to improved customer experience. Uncertainty can be caused by a change of shifts or lack of necessary equipment or information, needed for the task at hand. Modeling the assignment of personnel to clients as E-RAP

with unit costs ensures that the number of reassignments after edge failures is as small as possible.

Reservation Systems (V-RAP). Hotel managers try to operate at full capacity, while bookings are never immutable. People often cancel their trips due to change of personal plans or professional obligations. The common strategy to cope with reservation changes is overbooking. In this way it can be ensured that even after some of the arrangements are canceled, most or even all rooms are occupied. On the downside, this may result in providing accommodation for the overbooked clients elsewhere, causing additional expenses. Deciding on possible client-room assignments in the overbooking process while taking the overbooking costs into account can be modeled as V-RAP.

Infrastructure Hardening (RMAP). In production planning an allocation of orders to specific production lines is determined to be used in day-to-day business for a particular time horizon. Naturally, during operation this allocation is subject to uncertainty, e.g., the properties of the ordered product may be modified or maintenance can reduce the operational capability of a production line. In order to be resilient against possible incidents while meeting the obligations committed to the clients, it can be required to robustify the established setup without major interruptions. Robustification means to upgrade a production line in order to increase its versatility in terms of number of products that can be manufactured using this specific production line. The current setup corresponds to a fixed matching in a bipartite graph and this question can be modeled as RMAP providing the decision maker with guidance during the upgrading process. An additional constraint on the number of modifications after an incident can be imposed, reflecting the necessity not to perturb the running operation too much.

CONTRIBUTION AND OUTLINE

This thesis is organized around the three aforementioned optimization problems. Each of them is treated in a virtually self-contained chapter, which provides a detailed outline at the beginning. Cross-references are given where it might be helpful. We highlight the most important results here. The three main chapters are preceded by Chapter 2, where we present terminology, concepts and results from graph theory, complexity theory and robust optimization, which are used throughout this thesis. The chapter is concluded by giving an overview of related work from the literature.

Chapters 3 and 4 deal with the two variants of RAP. Unlike the majority of robust assignment problems treated in the literature (see Section 2.6.1), both problems studied here, capture structural uncertainty and thus provide novel insights for this classical combinatorial problem from the robust optimization point of view.

For both problems, E-RAP and V-RAP, we prove NP-hardness even in the case where failure scenarios are restricted to contain only one edge or node from the underlying bipartite graph. Moreover, both variants are as hard to approximate as Set Cover in this setting (Theorem 3.4.3). In particular this means an $(\ln n)$-approximation is the best we can hope for. Most characteristics are the same for both variants, but in two aspects E-RAP differs from V-RAP. First, deciding feasibility of an E-RAP instance where the scenario set is given implicitly is an NP-hard problem (see Section 3.2). Second, E-RAP remains NP-hard with only two singleton scenarios (see Theorem 3.3.3). This is quite surprising and makes E-RAP one of the few examples that a robust counterpart of a tractable nominal problem with a constant number of uncertain resources is NP-hard. For V-RAP both questions can be answered efficiently. The complexity results are completed by proving APX-hardness for both problems in the unit cost setting.

On the algorithmic side both optimization problems admit approximation algorithms matching the logarithmic lower bound asymptotically. The algorithm for E-RAP relies on randomized rounding techniques and the connection of feasible solutions to matching-covered graphs (see Section 3.5). The analog algorithm for V-RAP uses the classical greedy approximation for Set Cover applied to a subproblem.

Both chapters combined, provide a full coverage of bulk-robust assignment problems and complement the results in [ASZ15], where the authors studied bulk-robust counterparts of two other classical combinatorial problems, namely Shortest Path and Spanning Tree.

Chapter 5 deals with RMAP and focuses on computational complexity. In Section 5.2 a new, specialized version of the Matching Preclusion Number Problem is introduced to prove NP-hardness of the question of determining the robustness level of a fixed perfect matching. The main result of Chapter 5 is the proof, that RMAP is Set Cover-hard, even with single edge failures (see Theorem 5.3.3). Notably, the restriction of the recovery budget to very small numbers leads to a tractable problem via a reduction to Min-Cost Edge Cover (see Theorem 5.4.2).

Some supplementary results and examples were deferred to two appendices. In Appendix A an NP-completeness result used in Section 3.3 is proven and Appendix B contains some notes on issues arising when extending E-RAP to non-bipartite graphs.

2

PRELIMINARIES AND RELATED WORK

In this section, we first provide basic notation and concepts widely used throughout this document (Sections 2.1 to 2.5). Subsequently, in Section 2.6 we review results from the literature related to the optimization problems studied in this thesis.

2.1 BASIC NOTIONS FROM GRAPH THEORY

In this section we follow [Die00].

GRAPHS, ADJACENCY AND INCIDENCE

For a set S and an integer $k \in \mathbb{Z}_+$ we define $[S]^k := \{S' \subseteq S : |S'| = k\}$ and $[k] := \{1, \dots, k\}$. For a finite set V the tuple $G = (V, E)$ is called a simple graph if $E \subseteq [V]^2$ and $V \cap E = \emptyset$. We call the elements in V nodes and the elements in E edges. The node set of G is referred to as $V(G)$ and its cardinality as n. The edge set of G is referred to as $E(G)$ and its cardinality as m. All graphs in this document are simple unless stated otherwise and we just write graph.
Two nodes v, w are adjacent if $e = \{v, w\}$ is an edge of G. We say e connects v with w or e covers v and w. Two edges are adjacent if they share one node. The neighborhood of a set of nodes V' in G is defined as $N_G(V') := \{v \in V(G) \setminus V' : v \text{ is adjacent to a node } v' \in V'\}$. A node v is incident with an edge e and vice versa if $v \in e$. For a subset $V' \subseteq V$ we denote by $E(V')$ the set of all edges in E connecting two nodes in V'. For subsets $X, Y \subseteq V$ we denote by $E(X, Y)$ the set of all edges connecting a node in X with a node in Y in G. For an edge set $E' \subseteq E$ the set of all nodes incident with some edge in E' is denoted by $V(E')$. For a node v we denote by $\delta_G(v)$ the set of all edges incident with v in G. The degree of a node v is defined as $\deg_G(v) := |\delta_G(v)|$. A degree-zero node is called isolated. A graph is called k-regular if each node has degree k. A 3-regular graph is called cubic. In case the graph G is unambiguous the subscript is omitted.

SUBGRAPHS

A graph H is a subgraph of a graph G if $V(H) \subseteq V(G)$ and $E(H) \subseteq E(G)$. We then write $H \subseteq G$. If $V(H) = V(G)$ we say H spans G. A set $V' \subseteq V(G)$ induces the subgraph $(V', E(V'))$ denoted by $G[V']$.

A set $E' \subseteq E(G)$ induces the subgraph $(V(E'), E')$ denoted by $G[E']$. Given a graph $G = (V, E)$ and a node set V' we write $G - V'$ for the subgraph $G[V \setminus V']$. Given a set $E' \subseteq [V]^2$ we write $G - E'$ for the subgraph $(V, E \setminus E')$. The union of two graphs G and H is defined as the pairwise union of their node and edge sets and is denoted by $G + H$.

PATHS, CYCLES AND CONNECTIVITY

A subgraph P of a graph G defined by a sequence $(e_1, \ldots, e_k)$ of distinct edges in G such that each two subsequent edges are adjacent and only subsequent edges share nodes is called a path in G. Note that paths, as defined here, are always node-disjoint. Paths can also be defined by an appropriate sequence of nodes. We write $\text{tail}(P)$ for the first and $\text{head}(P)$ for the last node in the sequence. The nodes in $V(P) \setminus \{\text{tail}(P), \text{head}(P)\}$ are called internal nodes. We say the end nodes are connected by P. An edge sharing exactly one node with a path P is said to be incident with P. If $P = (e_1, \ldots, e_k)$ is a path and $k \geqslant 2$ then the sequence $(e_1, \ldots, e_k, \{\text{tail}(P), \text{head}(P)\})$ is called a cycle. The number of edges in a path P or a cycle C determines its length, i.e., $\text{length}(P) = |E(P)|$ and $\text{length}(C) = |E(C)|$. An edge connecting two non-adjacent nodes of a cycle is called a chord. We say a cycle or a path is odd if it has an odd number of edges and even otherwise. A graph G is k-node-connected if G contains k node-disjoint paths connecting any two of its nodes. Maximal connected subgraphs of G are called components.

BIPARTITE GRAPHS

A graph $G = (V, E)$ is called bipartite if its node set V can be partitioned in two sets U, W with $U \dot\cup W = V$ and $E \subseteq E(U, W)$. The tuple (U, W) is called the bipartition of G and is not unique in general. Hence, the bipartition is usually provided explicitly and we write $G = (U \dot\cup W, E)$. A bipartite graph is called balanced if $|U| = |W|$. A graph G is bipartite if and only if G has no odd cycles.

DIRECTED GRAPHS

For a finite set V the tuple $D = (V, A)$ is called directed graph or digraph if $A \subseteq V \times V$ and $V \cap A = \emptyset$. We call elements in V nodes and elements in A directed edges or arcs. For an arc $a = (v, w)$ we say a is directed from v to w. We refer to v as $\text{tail}(a)$ and to w as $\text{head}(a)$. A digraph is connected if the corresponding undirected graph is connected. A digraph is strongly connected if any two of its nodes are connected by a directed path. The remaining terms introduced for graphs can be defined accordingly for digraphs.

MISCELLANEOUS

A graph is called complete if every node is adjacent to every other

node in the graph. We write K_n for the complete graph on n nodes. In a bipartite graph $G = (U \dot\cup W, E)$ by completeness we mean that every node $u \in U$ is adjacent to every $w \in W$. We denote the complete graph on $n + m$ nodes by $K_{n,m}$.
An edge-weighted graph is a graph $G = (V, E)$ accompanied by a cost (or weight) function $c\colon E \to \mathbb{R}$. Given $F \subseteq E$, we write $c(F)$ or $\sum_{e\in F} c(e)$ for the cost of F. We write c_e instead of $c(e)$ and $c \in \mathbb{R}^E$ for brevity. Node-weighted graphs are defined analogously.

2.2 CONCEPTS AND NOTIONS FROM COMPLEXITY THEORY

In this thesis we are interested in the analysis of optimization problems in terms of computational complexity. Thus, in this section we briefly introduce the necessary definitions and concepts from complexity theory. For a more thorough introduction we refer to [GJ79] and [Aus+12].

A decision problem Π is specified by the set of legal inputs and a question, e.g., does a given graph have a node with degree at least 3? All legal inputs constitute the set of instances of Π. An algorithm $\mathcal{A}$ for Π is a finite sequence of instructions that performs operations on a given instance of Π and outputs either YES or NO correctly. We say $\mathcal{A}$ is a polynomial-time algorithm if, according to a computer model (usually a so-called Turing machine), its output is computed in time polynomially bounded by the size of the input. Such algorithms are called efficient.

Here we define the input size of a problem using the traditional binary encoding. This means for an integer z we have $\text{size}(z) = 1 + \lceil \log(|z| + 1) \rceil$. For a rational number we encode the numerator and denominator separately. For a vector or a matrix, the input size is the sum of the sizes of its elements. For a weighted graph $G = (V, G)$ with $c \in \mathbb{Q}^E$ the input size is $|V| + |E| + \text{size}(c) = n + m + \sum_{e\in E} \text{size}(c_e)$. We denote P as the set of all decision problems admitting an efficient algorithm. Problems in P are called tractable.

There are problems not known to be tractable, some are even known to be undecidable. Natural candidates for tractable decision problems are problems with efficient certifiers for instances with a positive answer. An efficient certifier $\mathcal{A}$ for a problem Π is a polynomial-time algorithm that takes two arguments: an instance $\mathcal{I}$ of Π and a string c. Additionally $\mathcal{A}$ has the following property. The instance $\mathcal{I}$ is a YES-instance if and only if there exists a certificate c with size bounded by $\text{size}(\mathcal{I})$ and $\mathcal{A}$ outputs YES for the input $(\mathcal{I}, c)$. The string c is called a short certificate for $\mathcal{I}$. We denote the set of such problems by NP. For any tractable problem Π we can use an empty string as the certificate and any polynomial-time algorithm for Π as a certifier, consequently

$\mathsf{P} \subseteq \mathsf{NP}$. However, it is not known if $\mathsf{P} = \mathsf{NP}$ but the assumption $\mathsf{P} \neq \mathsf{NP}$ is widely expected to be true.

A complement of a decision problem results from swapping roles of Yes and No answers. Complements of problems in NP form the complexity class coNP. In other words, problems in coNP admit efficient certifiers for No-instances.

To classify problems concerning their tractability we use the concept of reductions. Reductions are functions that transform a given instance $\mathcal{I}$ of a problem Π into an instance $\mathcal{I}'$ of a different problem Π'. We call a reduction f an NP-reduction if it is polynomial-time computable and answer-preserving, i.e., $\mathcal{I}$ is a Yes-instance of Π if and only if $f(\mathcal{I}) = \mathcal{I}'$ is a Yes-instance of Π'. NP-reductions are also called many-to-one reductions in the literature. The definition above immediately implies that if Π' is tractable so is Π.

We say a problem Π is NP-hard if each problem in NP is NP-reducible to Π. In other words, NP-hard problems are at least as hard as any other problem in NP. We say a problem is NP-complete if it is NP-hard and it is contained in the class NP. NP-hardness provides strong evidence that a problem is intractable. Because NP-reductions are transitive we can prove NP-hardness by providing an NP-reduction from an NP-complete problem.

2.2.1 Optimization Problems

However, most problems encountered in real life can not be reduced to a yes or no question because they have several acceptable answers that can be measured according to their quality. The mathematical equivalent of such problems are optimization problems, which we define next.

Definition 2.2.1. An optimization problem Π is defined by a tuple $(\mathfrak{I}, \texttt{sol}, \texttt{val}, \texttt{type})$, where:

(a) $\mathfrak{I}$ is the set of instances of Π;

(b) $\texttt{sol}(\mathcal{I})$ denotes the set of feasible solutions to an instance $\mathcal{I} \in \mathfrak{I}$;

(c) the function $\texttt{val}$ defines, given $\mathcal{I} \in \mathfrak{I}$ and $X \in \texttt{sol}(\mathcal{I})$, the non-negative rational objective value[1] of X with respect to $\mathcal{I}$;

(d) $\texttt{type} \in \{\min, \max\}$ describes whether Π is a minimization or a maximization problem.

If the referenced problem Π is not evident from the context, we write $\mathfrak{I}_\Pi$ and so forth.

1 It is more natural to use real numbers when describing problems, but there is no finite binary encoding for arbitrary reals. Hence, when dealing with irrational values we have to approximate them using rationals. This can be done within any given error specified by a rational $\varepsilon > 0$ [GLS93, pp. 33-36].

The goal of an optimization problem Π, given an instance $\mathcal{I}$ of Π, is to determine the best possible value

$$\texttt{type}\,\{\texttt{val}(\mathcal{I}, X) \colon X \in \texttt{sol}(\mathcal{I})\}$$

and a corresponding solution X. We denote the optimal value by $\texttt{val}^*(\mathcal{I})$. For an optimal solution for an instance $\mathcal{I}$ we write $\mathrm{OPT}(\mathcal{I})$. We denote by PO the set of optimization problems such that $\mathrm{OPT}(\mathcal{I})$ and $\texttt{val}^*(\mathcal{I})$ can be computed efficiently, i.e., by an algorithm with polynomially bounded running time. Such algorithms are called efficient or exact.

We can derive a decision problem from an optimization problem Π by providing a bound on the objective and asking whether this bound is exceeded by any feasible solution. We denote the resulting decision problem by Π_D. In order to relate optimization problems to decision problems in NP we impose additional requirements: For every (not necessarily legal) input $\mathcal{I}$, X it is possible to check efficiently if $X \in \texttt{sol}(\mathcal{I})$ and if it is the case, to efficiently compute $\texttt{val}(\mathcal{I}, X)$. Optimization problems satisfying these extra requirements form the class NPO. By definition, for every optimization problem $\Pi \in \mathrm{NPO}$, its decision variant Π_D is contained in NP.

We can show NP-hardness of an NPO problem Π' by providing an NP-reduction from an NP-complete problem to Π_D. The other possibility to prove NP-hardness of an NPO problem Π' is the provision of a polynomial-time reduction $f \colon \mathfrak{I}_\Pi \to \mathfrak{I}_{\Pi'}$ from some NP-hard decision or optimization problem Π such that f can be used to define an exact algorithm for Π using a Π'-oracle on $f(\mathcal{I})$. A Π'-oracle is a function that returns an optimal solution and the optimal objective value of a given instance $\mathcal{I}' \in \mathfrak{I}_{\Pi'}$ in constant time. Reductions of this type are called Turing reductions.

The classification of solutions to an optimization problem into optimal and non-optimal solutions is sufficient, but too coarse for many purposes. In the latter case we also could be comfortable with solutions that are acceptable in terms of its objective value. The concept of reasonably good solutions is formalized in the next definition. From now on we restrict ourselves to NPO minimization problems, the definitions for maximization case are analogously.

Definition 2.2.2. Let Π be an NPO minimization problem with a non-negative objective function. An approximation algorithm $\mathcal{A}$ for Π is a finite sequence of instructions that performs operations on a given instance $\mathcal{I} \in \mathfrak{I}_\Pi$ and terminates after a number of steps polynomially bounded by $\mathrm{size}(\mathcal{I})$. Its output $\mathcal{A}(\mathcal{I})$ is a solution to $\mathcal{I}$ and there exists a function $r \colon \mathbb{Z}_+ \to [1, \infty)$ such that

$$\texttt{val}(\mathcal{A}(\mathcal{I})) \leqslant r(\mathrm{size}(\mathcal{I})) \cdot \texttt{val}^*(\mathcal{I})$$

holds for all $\mathcal{I} \in \mathfrak{I}_\Pi$. The function r is the approximation ratio of $\mathcal{A}$.

Usually we simply write approximation. An efficient (exact) algorithm is an approximation with $r = 1$.

An approximation has an arbitrary precision if for any fixed $\varepsilon > 0$ the approximation ratio is $(1 + \varepsilon)$. We distinguish two major cases concerning the behavior of a approximation with respect to the additional precision parameter ε. An approximation with running time $O(\text{size}(\mathcal{I})^{f(1/\varepsilon)})$ for some function f is called a PTAS. For every fixed ε a PTAS has polynomial running time but the exponent can be very large for small ε (e.g., $f = \exp$ satisfies the definition). In contrast, an FPTAS is required to have running time $O((1/\varepsilon)^{O(1)} \cdot \text{size}(\mathcal{I})^{O(1)})$. Hence the difference is how we trade time for accuracy. The classes of all optimization problems admitting a PTAS or FPTAS are denoted by the same name.

The set of optimization problems admitting a constant factor approximation is denoted by APX, i.e., these problems have approximations such that the ratio function r is constant. The prevailing assumption $P \neq NP$ implies

$$PO \subsetneq FPTAS \subsetneq PTAS \subsetneq APX \subsetneq NPO.$$

From now on we usually refer to NPO optimization problems as problems.

2.2.2 Approximation-Preserving Reductions

Reductions described in the previous section are not powerful enough to derive positive or negative approximability results for problems in NPO. NP-reductions are not required to be constructive, hence it is not always possible to provide an actual solution to the instance the reduction was applied to. The problem with Turing reductions is that we do not have any control of the cost of approximate solutions.

The property of a reduction to be constructive is formalized next. Then we define two specialized approximation-preserving reductions that are used in this thesis.

Definition 2.2.3 (basic reduction)**.** Given two NPO optimization problems Π and Π', the tuple (f, g) is called a basic reduction from Π to Π' if:

(B1) Function f maps instances of problem Π to instances of problem Π', i.e., $f\colon \mathcal{I}_\Pi \to \mathcal{I}_{\Pi'}$;

(B2) Function f upholds feasibility, i.e., if $\mathcal{I} \in \mathcal{I}_\Pi$ is feasible then $f(\mathcal{I}) \in \mathcal{I}_{\Pi'}$ is feasible as well;

(B3) Given $\mathcal{I} \in \mathcal{I}_\Pi$, function g maps solutions to $f(\mathcal{I})$ to solutions to $\mathcal{I}$, i.e., for each $X' \in \texttt{sol}(f(\mathcal{I}))\colon g(\mathcal{I}, X') \in \texttt{sol}(\mathcal{I})$;

(B4) Both functions f, g are polynomial-time computable.

A basic reduction is still not powerful enough because we do not know anything about the objective values of solutions to $f(\mathcal{I})$ and its counterparts constructed by g. What we need is a guarantee that "good" solutions to Π' are mapped to "good" solutions to Π.

Definition 2.2.4 ([PY91]). Let (f, g) be a basic reduction from Π to Π'. If two positive constant parameters α and β exist such that

(L1) For any $\mathcal{I} \in \mathfrak{I}_\Pi$: $\mathtt{val}^*_{\Pi'}(f(\mathcal{I})) \leqslant \alpha\, \mathtt{val}^*_\Pi(\mathcal{I})$,

(L2) For any $\mathcal{I} \in \mathfrak{I}_\Pi$ and for any $X' \in \mathtt{sol}(f(\mathcal{I}))$ there holds

$$\left|\mathtt{val}^*_\Pi(\mathcal{I}) - \mathtt{val}_\Pi(\mathcal{I}, g(\mathcal{I}, X'))\right| \leqslant \beta \left|\mathtt{val}^*_{\Pi'}(f(\mathcal{I})) - \mathtt{val}_{\Pi'}(f(\mathcal{I}), X')\right|,$$

then the tuple (f, g, α, β) is called an L-reduction.

L-reductions relate the optimal values as well as the absolute errors of approximate solutions to the two problems linearly. Given an approximation algorithm $\mathcal{A}'$ for Π' we can deduce an approximation $\mathcal{A}(\mathcal{I}) := g(\mathcal{I}, \mathcal{A}'(f(\mathcal{I})))$ for Π. Combining (L1) and (L2) gives

$$\frac{\mathtt{val}_\Pi(\mathcal{I}, g(\mathcal{I}, X')) - \mathtt{val}^*_\Pi(\mathcal{I})}{\mathtt{val}^*_\Pi(\mathcal{I})} \leqslant \alpha\beta \left(\frac{\mathtt{val}_{\Pi'}(f(\mathcal{I}), X') - \mathtt{val}^*_{\Pi'}(f(\mathcal{I}))}{\mathtt{val}^*_{\Pi'}(f(\mathcal{I}))}\right).$$

Plugging the definition of $\mathcal{A}$ into the latter inequality implies that if $\mathcal{A}'$ is a r'-approximation for Π', then $\mathcal{A}$ is a $(1 + \alpha\beta(r' - 1))$-approximation for Π. Hence if $\mathcal{A}'$ is a PTAS, so is $\mathcal{A}$. On the other hand if we know that Π cannot be approximated to within a certain bound the same is certainly true for Π'. We use L-reductions to prove APX-hardness of problems using the following property: If a problem Π is APX-complete and L-reducible to a problem Π', then Π' is APX-hard [Aus+12, Lem. 8.2][2]. APX-hard problems are the analog of NP-hard problems for the class APX, i.e., these problems are the hardest ones in APX.

To conclude this subsection we present a second type of an approximation-preserving reduction used in this document.

Definition 2.2.5 ([CFS91]). A basic reduction (f, g) from Π to Π' is an S-reduction[3] if the following two properties hold.

(S1) For every $\mathcal{I} \in \mathfrak{I}_\Pi$ and for any $X' \in \mathtt{sol}(f(\mathcal{I}))$ there holds

$$\mathtt{val}_\Pi(\mathcal{I}, g(\mathcal{I}, X')) = \mathtt{val}_{\Pi'}(f(\mathcal{I}), X').$$

(S2) For every $\mathcal{I} \in \mathfrak{I}_\Pi$: $\mathtt{val}^*_\Pi(\mathcal{I}) = \mathtt{val}^*_{\Pi'}(f(\mathcal{I}))$.

Every S-reduction is an L-reduction with $\alpha = \beta = 1$. S-reductions transfer all inapproximability results from Π to Π'. Because of this they are sometimes referred to as "strong" reductions.

2 This implies that Π' is APX-hard with respect to the AP-reducibility (see [Aus+12, Def. 8.3]) which is commonly used to define APX-hardness (see [Aus+12, Def. 8.5]). Since L-reductions are easier to handle, we omit further details here and do not use AP-reductions explicitly.

3 The concept of S-reducibility is different from the notion of strict reducibility introduced in [OM87]. Every S-reduction is a strict reduction, but not vice versa.

2.2.3 Parameterized Complexity

As discussed in the beginning of this chapter NP-hard problems are unlikely to have polynomial-time exact algorithms. For optimization problems one approach dealing with this issue is to relax the condition on the quality of the returned solution. This approach leads to approximation algorithms and was presented in the previous subsection. Here we introduce a different concept. The core idea is to loosen the polynomiality condition on the running time while still demanding exact solutions. For a detailed introduction to parameterized complexity consult [Cyg+15].

We are going back to decision problems and address the question which structural properties make a problem hard. For this reason we introduce the notion of parameterization. Let $\Pi \in \mathsf{NP}$. In addition to the input size n we introduce a function $k\colon \mathfrak{I}_\Pi \to \mathbb{Z}_+$ called the parameter of Π. The tuple (Π, k) is called a parameterized decision problem. The parameter k reflects some structural properties of Π. A natural choice is $k = \mathtt{val}^*_\Pi$ and we then call (Π, k) the standard parameterization of Π. For graph problems maximum degree or treewidth, a measure of a graph's similarity to a tree, could be of interest.

A parameterized algorithm $\mathcal{A}$ for a problem (Π, k) is an exact algorithm with running time $f(k) \cdot n^{g(k)}$, where $f,g\colon \mathbb{Z}_+ \to \mathbb{Z}_+$ are computable functions. The class of problems admitting such algorithms is called XP. Algorithms of this kind are called slice-wise polynomial or simply XP-algorithms. If $g = O(1)$, i.e., it is a constant independent of k and n, we say the algorithm is a fixed-parameter algorithm. Parameterized problems admitting a fixed-parameter algorithm are called fixed-parameter tractable and constitute the class FPT. Hence FPT can be seen as the analog of P in the world of parameterized problems.

The classification of problems is again obtained using reductions. We can use NP-reductions as a starting point, but they are not strong enough[4]. A parameterized reduction from (Π, k) to (Π', k') is answer-preserving and runs in time $f(k) \cdot n^{O(1)}$. Additionally there is a computable function b such that $k' \leqslant b(k)$, that is b controls the growth of the parameter k'. The S-reduction as presented in Definition 2.2.5 is parameterized reduction with respect to the standard parameter solution size. Unlike for decision problems there is a hierarchy of classes of parameterized decision problems. The definition is based on Boolean circuit complexity and these classes are denoted by $W[t]$, $t \geqslant 0$ [Cyg+15, Sec. 13.3]. The class corresponding to NP is $W[1]$. Assuming the exponential time hypothesis[5] we have that

$$\mathsf{FPT} = W[0] \subsetneq W[1] \subsetneq W[2] \subsetneq \cdots \subsetneq \mathsf{XP}.$$

4 The classical reduction from Stable Set to Node Cover (see p. 21) is not a parameterized reduction. Moreover, Stable Set is W[1]-complete but Node Cover $\in$ FPT.

5 The ETH states that 3-SAT can not be solved in subexponential time, i.e., $O(2^{o(n)})$.

If a problem admits an FPTAS, then its standard parameterization belongs to FPT [DF99, Thm. 4.3]. An example of a parameterized problem that is $W[t]$-complete for some $t \geqslant 1$ is the standard parameterization of Set Cover (see Section 2.4).

2.3 MATCHINGS IN BIPARTITE GRAPHS

We focus on matchings in bipartite graphs here, for more details consult [LP86].

Two edges in a graph G are independent if they are non-adjacent. A matching M in G is a set of independent edges. We say a node v is covered or saturated by M if v is incident with one of the edges in M. Nodes that are not covered are called exposed or free. A matching with no exposed nodes is a perfect matching. For a set S of nodes we say M is an S-perfect matching if all nodes in S are covered. The number of perfect matchings can be very large. Moreover, counting perfect matchings in bipartite graphs is an NP-hard problem [Val79]. Fukuda and Matsui [FM94] provided an algorithm to compute all perfect matchings in a bipartite graph G in time $O(N(n+m)+n^{2.5})$, where N is the number of perfect matchings in G.

Bipartite graphs with U-perfect matchings can be characterized in the following way.

Theorem 2.3.1 (Hall). *A bipartite graph $G = (U \dot\cup W)$ contains a U-perfect matching if and only if*

$$\forall\, T \subseteq U\colon |T| \leqslant |N(T)|.$$

Matchings of maximum cardinality in G are called maximum matchings and their size is denoted by $\nu(G)$.

The general matching problem is defined as follows.

Problem 2.3.2 (MWMP).

Maximum-Weight Matching	
Instance:	$\langle G, w\rangle$, where G is a graph and $w \in \mathbb{R}^{E(G)}$ weights.
Solution:	A matching $M \subseteq E(G)$.
Task:	Find M maximizing $w(M)$.

In case $w = \mathbb{1}$, solutions to MWMP are maximum matchings. Hall's theorem cannot be exploited algorithmically directly, instead matchings can be constructed iteratively as we describe next.

Let G be a graph and M a matching in G. A path P in G is called M-alternating if $E(P) \setminus M$ is a matching. The path P is called M-augmenting if both of its end nodes are exposed by M. Analogously,

a cycle C is M-alternating if $E(C) \setminus M$ is a matching. Instead of M-alternating path or cycle we sometimes just say M-path or M-cycle.

If M is a matching in G and P an M-augmenting path, then the symmetric difference $M \triangle P := (M \setminus E(P)) \cup (E(P) \setminus M)$ is a matching of size $|M| + 1$. M-alternating paths can be found using depth-first search (DFS). The following characterization is fundamental in matching theory.

Theorem 2.3.3 (Berge)**.** *Let G be a graph and M a matching in G. Then M is a maximum matching if and only if the graph G has no M-augmenting path.*

Berge's Theorem is exploited successfully by several algorithms. The Hungarian Method [Kuh55] is the oldest matching algorithm. It determines a minimum-cost perfect bipartite matching in time $O(n^4)$. Hopcroft and Karp [HK73] provided an $O(\sqrt{n}(n+m))$ algorithm to find a maximum matching in a bipartite graph. In non-bipartite graphs maximum matchings can be found using the famous Blossom Algorithm [Edm65]. All three algorithms can be improved in terms of running time using more sophisticated data structures and subroutines.

The notion of perfect matchings can be generalized in the following manner. An edge set F of a graph G is a k-factor if the subgraph $G[F]$ is spanning and k-regular. A perfect matching is a 1-factor. If $G[F]$ is spanning and each node v in $V(G)$ has $\deg_{G[F]} = f(v)$, where $f\colon V(G) \to \mathbb{Z}_+$, then F is called an f-factor. Finding f-factors can be reduced to finding perfect matchings in a transformed graph [Tut54]. In bipartite graphs a k-factor can be decomposed into k disjoint perfect matchings.

2.3.1 The Matching Polytope

In this section we derive a description of the matching polytope. For an introduction to linear and integer programming theory we refer the reader to [Sch98].

Consider a graph G. For a set $S \subseteq E(G)$ the incidence vector of S is defined as

$$\chi^S_e := \begin{cases} 1, & e \in S, \\ 0, & e \notin S. \end{cases}$$

The matching polytope for a graph G is defined as

$$M(G) := \operatorname{conv}\{\chi^M : M \text{ is a matching in } G\},$$

where conv denotes the convex hull of a set of vectors. Using $M(G)$ we can restate Problem 2.3.2 as $\max\{w^\top \chi^M : M \in M(G)\}$. Observe

that for a graph $G = (V, E)$ this problem is equivalent to the following integer linear program

$$\begin{aligned} \max \quad & w^\top x \\ \text{s.t.} \quad & x(\delta(v)) \leqslant 1 \qquad \text{for each } v \in V, \\ & x \in \{0,1\}^V. \end{aligned}$$

This ILP can be described using the node-edge incidence matrix A of graph G. The matrix A is indexed by nodes and edges of G and defined as

$$A_{ve} := \begin{cases} 1, & v \in e, \\ 0, & v \notin e. \end{cases}$$

For a bipartite graph the incidence matrix A is totally unimodular (TU) and we can drop the integrality constraints and use the LP relaxation, i.e.,

$$M(G) = \operatorname{conv}\{x \in \mathbb{Z}_+^E : Ax \leqslant \mathbb{1}\} = \{x \in \mathbb{R}_+^E : Ax \leqslant \mathbb{1}\}.$$

Thus we can restate Problem 2.3.2 as the following LP

$$\max \{w^\top x : Ax \leqslant \mathbb{1},\ x \in \mathbb{R}_+^E\}. \tag{1}$$

Linear programs can be solved in time polynomial in the input size of A and w using the ellipsoid method [Kha80]. Unlike the simplex algorithm the ellipsoid method does not necessarily return an optimal vertex solution to a linear program. Fortunately this issue can be overcome, for details see [GLS93, Rem. 6.5.2]. Khachiyan's result can be strengthened by removing the ellipsoid method's dependence on the cost function and the right hand side vector [Tar86]. This means, we can solve the LP in (1) in strongly polynomial time.

2.3.2 Matching-Covered Graphs

An edge of a graph G is called allowed if it is contained in some perfect matching of G. We say an edge that is not part of any perfect matching is dispensable. A graph G is matching-covered if each of its edges is allowed. Note that here, unlike in standard literature, we do not impose any connectivity requirements on the graph. In [LP86] matching-covered graphs are always connected and called elementary. We will see later, in Chapter 3, that solutions to E-RAP can be described in terms of matching-covered graphs and repeatedly exploit this property algorithmically.

Matching-covered graphs are well-studied in graph theory and have plenty of characterizations as the following theorem indicates.

Theorem 2.3.4 ([LP86, Thm. 4.1.1][6]). *For a bipartite graph* $G = (U \dot\cup W, E)$ *the following statements are equivalent.*

(a) G *is connected and matching-covered.*

(b) Either $G = K_2$ *or* $|V(G)| \geqslant 4$ *and for any* $u \in U$ *and* $w \in W$ *the subgraph* $G - \{u, w\}$ *has a perfect matching.*

(c) For each $\emptyset \neq T \subsetneq U$ *there holds* $|T| + 1 \leqslant |N_G(T)|$.

Whitney [Whi32] provided a characterization of 2-node-connected graphs based on path decompositions. Matching-covered graphs have similar properties. In order to describe these decompositions we need some new vocabulary.

Let $G = (U \dot\cup W, E)$ be a bipartite graph, and let $H \subseteq G$ be a subgraph. An ear in G with respect to H is an odd path P in G that connects a node in U with a node in W. Additionally the internal nodes of P are contained in $V(G) \setminus V(H)$.

Definition 2.3.5. An ear decomposition of a connected bipartite graph G is a sequence $P_0, P_1, \ldots, P_q$ of paths in G with the following properties.

- The path P_0 has length one, i.e., it is a single edge.
- For every $j \in [q]$, the path P_j is an ear with respect to the graph $H_{j-1} := P_0 + \ldots + P_{j-1}$;
- There holds $G = P_0 + \ldots + P_q$.

An example of an ear decomposition can be found in Fig. 1. An ear

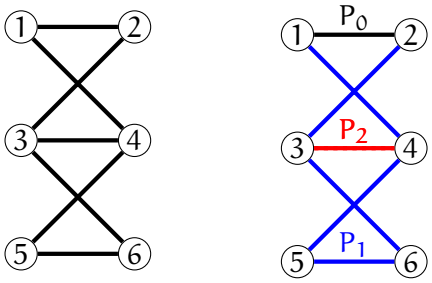

Figure 1: A bipartite graph (left) and its ear decomposition (right).

decomposition is by far not unique, but the number of ears for a connected graph G is always $q = |E(G)| - |V(G)| + 1$. This is the cyclomatic number of G, i.e., the minimum number of edges to be removed from G in order to destroy every cycle in G. Ear decompositions gives us an additional characterization of connected matching-covered graphs as we see next.

6 According to [LP86], this result is due to Hetyei [Het64].

Theorem 2.3.6 ([LP86, Thm. 4.1.6][7]). *Let G be a connected bipartite graph. Then, G is matching-covered if and only if G has an ear decomposition.*

This result implies that connected matching-covered graphs can be constructed from any edge $e \in E$ by a sequence of ear additions. Moreover, we can deduce that adding an edge from U to W to G yields again a matching-covered graph. Theorem 2.3.6 also implies that matching-covered graphs are always 2-node-connected.

Note that the graphs H_{j-1} from Definition 2.3.5 have the property that the nodes $V(G) \setminus V(H_{j-1})$ are perfectly matchable. We call subgraphs possessing this property nice. The graphs H_{j-1} are nice because an ear decomposition uniquely defines a perfect matching M such that all ears are M-alternating. In Fig. 1 the matching M is formed by the vertical edges.

In order to explain how to compute an ear decomposition efficiently, we need the following auxiliary graph.

Definition 2.3.7. Let M be a perfect-matching in $G = (U \dot\cup W, E)$. We denote by $D(M, G)$ a digraph on the node set $V(G)$ that is defined as follows. We orient the edges of M from U to W and the edges $E(G) \setminus M$ from W to U (see Fig. 2).

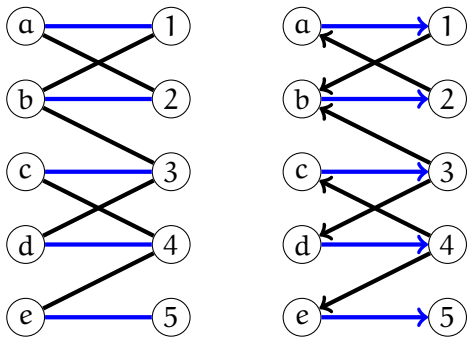

Figure 2: A graph with a perfect matching M (blue) and the corresponding digraph $D(G, M)$.

The digraph $D(G, M)$ has several useful properties. First observe that directed paths in $D(G, M)$ are M-alternating. Hence, we can use this digraph to compute an ear decomposition of G using a folklore algorithm stated in Algorithm 1. Note that the most expensive step of Algorithm 1 is the computation of a perfect matching.

The digraph $D(G, M)$ has additional useful properties. Directed cycles in $D(G, M)$ are M-alternating and nice. Moreover, $D(G, M)$ is acyclic if and only if the perfect matching M is unique. Furthermore

7 According to [LP86], this result is due to Hartfiel [Har70] and was presented in the context of decomposition of matrices.

Algorithm 1: Bipartite Ear Decomposition Algorithm

Input: Connected matching-covered graph $G = (U \dot\cup W, E)$
Output: An ear decomposition of G

$M \leftarrow$ a perfect matching in G
$D \leftarrow D(G, M)$ as defined in Definition 2.3.7
$P_0 \leftarrow$ an edge in G
$H \leftarrow P_0$
for $i \in [q]$ **do**
  $U', W' \leftarrow$ the bipartition of H induced by U, W
  $\hat{P}_i \leftarrow$ a directed path from W' to U' in D with internal nodes in $V(D) \setminus V(H)$
  $P_i \leftarrow$ undirected version of $\hat{P}_i$
  $H \leftarrow H + P_i$
end
return $(P_0, \ldots, P_q)$

the digraph $D(G, M)$ is strongly connected if and only if G is connected and matching-covered [LP86, p. 123], which leads to the following observation.

Observation 2.3.8. *Let M be a perfect-matching in $G = (U \dot\cup W, E)$, $D := D(G, M)$ and $\vec{M}$ the arcs in D corresponding to M. Then, the graph G is matching-covered if and only if all arcs in $A(D) \setminus \vec{M}$ are contained in the strongly connected components of the digraph D.*

Hence we can use the digraph $D(G, M)$ to identify all dispensable edges in G. For example in the graph presented in Fig. 2 the edges $\{b, 3\}$ and $\{e, 4\}$ are dispensable. This algorithmic idea is formalized and analyzed in [Rég94, Alg. 2]. The running time of the algorithm is dominated by the time needed to compute a perfect matching because detecting strongly connected components is essentially a pass of DFS, which has running time $O(n + m)$. Note that the removal of dispensable edges disconnects the graph. Identifying edges belonging to any minimum-cost perfect matching is possible using dual optimal solutions to Maximum-Weight Matching [VV16].

The notion of matching-covered graphs can be generalized: A graph is called k-extendable if every matching of size k is contained in a perfect matching. Matching-covered graphs are hence 1-extendable. A characterization of n-extendable graphs similar to Theorem 2.3.4 is possible.

Theorem 2.3.9 ([Plu86, Thm. 2.2][8]). *Let $G = (U \dot\cup W, E)$ be a bipartite connected graph and $k \leqslant \frac{|V(G)|-2}{2}$. Then the graph G is k-extendable if and only if $|U| = |W|$ and*

$$\forall\, \emptyset \neq T \subsetneq U\colon |T| + k \leqslant |N_G(T)|.$$

8 According to [Plu86] this result is due to Brualdi and Perfect [BP71].

We will encounter this condition again when we will discuss feasibility of V-RAP (see p. 69). Lakhal and Litzler [LL98] provided an algorithm to determine the maximal extendability $\bar{k}$ of a graph G in time $O(m \cdot \min\{\bar{k}^3 + n, \bar{k} \cdot n\})$. For an overview on results for n-extendable graphs we refer to the survey of Plummer [Plu94].

It is worth mentioning that the notion of matching-covered graphs can also be defined for non-bipartite graphs but the situation is more complicated. One of the highlights is the Two Ear Theorem [LP86, Thm. 5.4.6]. The best known algorithm for computing non-bipartite ear decompositions can be found in [CC05].

2.4 A BRIEF INTRODUCTION TO SET COVER

SET COVER is one of the 21 NP-complete decision problems from Karp's seminal paper [Kar72]. We use its inapproximability and algorithmic results throughout this document and for this reason repeat the most important results here. The problem is defined in Problem 2.4.1 and an example is illustrated in Fig. 3.

Problem 2.4.1 (SCP).

SET COVER	
Instance:	$\langle [k], \mathcal{S} \rangle$, where $[k]$, $k \in \mathbb{Z}_+$, is a ground set and $\mathcal{S}$ a collection of nonempty subsets of $[k]$.
Solution:	Cover $\mathcal{C}$, i.e., $\mathcal{C} \subseteq \mathcal{S}$ with $[k] = \bigcup_{S \in \mathcal{C}} S$.
Task:	Find a cover minimizing $\lvert\mathcal{C}\rvert$ or decide that no cover exists.

Throughout this document we follow the standard assumption that the size of the collection $|\mathcal{S}| = \ell$ is bounded by a polynomial in k. Hence, we usually refer to k as the size of a SET COVER instance.

We can neglect infeasible instances of SCP in our analysis. For an instance $\langle [k], \mathcal{S} \rangle$ we can efficiently verify if $\mathcal{S}$ itself is a cover by checking $\bigcup_{S \in \mathcal{S}} S = [k]$. In case of success, we call $\mathcal{S}$ a trivial cover. The weighted version of SCP, where an instance additionally provides a cost function $c \in \mathbb{R}_+^{\mathcal{S}}$, is denoted by WEIGHTED SET COVER (WSCP).

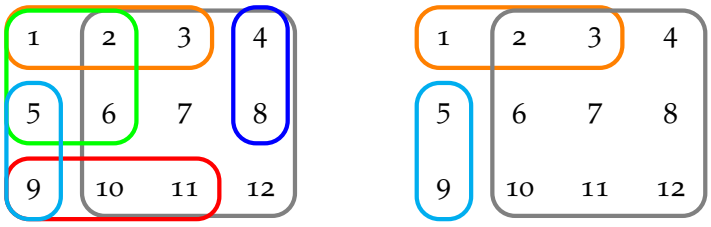

Figure 3: An instance of SCP (left) and an optimal solution (right).

Both variants of Set Cover are notoriously hard to approximate. A common approach is to use the greedy algorithm described in Algorithm 2.

Algorithm 2: Classical greedy algorithm for WSCP

Input: A feasible Weighted Set Cover instance $\langle [k], \mathcal{S}, c \rangle$
Output: A cover for $[k]$

$\mathcal{C} \leftarrow \emptyset$
$U \leftarrow [k]$ // uncovered elements
while $U \neq \emptyset$ **do**
  $S^* \leftarrow \arg\max \left\{ \frac{|S \cap U|}{c(S)} : S \in \mathcal{S} \setminus \mathcal{C} \right\}$
  $\mathcal{C} \leftarrow \mathcal{C} \cup \{S^*\}$
  $U \leftarrow U \setminus S^*$
end
return $\mathcal{C}$

For unweighted Set Cover the greedy algorithm was first presented and analyzed by Johnson [Joh74] and Lovász [Lov75] independently. Both authors argued that the approximation guarantee of the greedy algorithm is $1 + \frac{1}{2} + \cdots + \frac{1}{r}$, where $r := \max_{S \in \mathcal{S}} |S|$ and is obviously bounded by k. This expression is also known as the r-th harmonic number, which is bounded by $\ln r + 1$. Chvátal extended these results to Weighted Set Cover.

Theorem 2.4.2 ([Chv79]). *Algorithm 2 is an* $(\ln k + 1)$*-approximation for* WSCP.

Later, Slavík [Sla96] provided a tight analysis for the greedy algorithm. He proved that its approximation ratio is in fact $\ln k - \ln \ln k + \Theta(1)$.

The first lower bound on the approximation quality was proved by Lund and Yannakakis [LY94]. They showed that, unless $\mathsf{NP} \subseteq \mathsf{DTIME}(n^{\mathrm{poly}\log n})$, SCP does not admit an approximation with better performance than $\frac{1}{4} \log k$.[9] The next big step was the celebrated quasi-NP-hardness result by Feige.

Theorem 2.4.3 ([Fei98]). *For every* $\varepsilon > 0$, Set Cover *admits no polynomial time* $(1 - \varepsilon) \ln k$*-approximation unless* $\mathsf{NP} \subseteq \mathsf{DTIME}(n^{\log\log n})$.

Recently the condition in Feige's result was weakened to $\mathsf{P} \neq \mathsf{NP}$ by Dinur and Steurer.

Theorem 2.4.4 ([DS14]). *For every* $\varepsilon > 0$, *it is* NP*-hard to approximate* Set Cover *to within* $(1 - \varepsilon) \ln k$.

9 A polylogarithmic function has running time of $O((\log n)^c)$ for some constant c. The statement $\mathsf{NP} \subseteq \mathsf{DTIME}(n^{\mathrm{poly}\log n})$ would imply the existence of quasi-polynomial time algorithms for NP-complete problems.

Their contribution was the last in a series of works on the techniques used in Feige's proof. For more details on other publications in this series consult [DS14]. This result means that the greedy algorithm is essentially the best we can hope for.

What are the structural properties that make Set Cover a hard problem? We introduce two parameters to describe the structure of an SCP instance $\langle [k], \mathcal{S} \rangle$. Assume that every element $i \in [k]$ occurs in the sets of the collection $\mathcal{S}$ the same number of times. This defines the first parameter, which we call a. If the size of each set in the collection coincide, then this number is our second parameter b. Class of SCP instances with parameters a and b is denoted by (a, b)-SCP. The set of instances parameterized only by a is denoted as $(a, *)$-SCP. An instance with $a = 1$ is trivially solvable because there is no choice to make. To discuss the case $a = 2$ we use a graph problem related to Set Cover.

Problem 2.4.5 (NCP)**.**

Node Cover	
Instance:	$\langle G \rangle$, where G is a graph.
Solution:	Node cover V', i.e., $V' \subseteq V(G)$ such that every edge in G is incident with at least one node in V'.
Task:	Find a node cover minimizing $\lvert V' \rvert$.

The relationship between $(2, *)$-SCP and NCP is presented in the next observation.

Observation 2.4.6. *We can equivalently reformulate* $(2, *)$-SCP *as* Node Cover *and vice versa. Given an* $(2, *)$-SCP *instance* $\langle [k], \mathcal{S} \rangle$*, the graph* G *for* NCP *is defined as follows. We set*

$$V := \mathcal{S} \text{ and } E := \{\{S, S'\} \colon S \cap S' \neq \emptyset,\ S \neq S'\}.$$

The parameter b *acts as the bound on the maximum degree of* G*.*

Node Cover is a classical combinatorial problem and is known to be NP-hard [Kar72]. The threshold on tractability for (a, b)-SCP can be derived from results for Node Cover in regular graphs.

Theorem 2.4.7 ([AK00],[CC03])**.** Node Cover *in cubic graphs is* APX*-complete and* NP*-hard to approximate to within* $\frac{100}{99}$*.*

For Node Cover in cubic graphs we write 3-NCP. Note that if the degree is at most 2, then Node Cover can be solved in polynomial time. This means (2,3)-SCP is the most restricted subclass of Set Cover instances remaining NP-hard.

We finish this subsection with a well-known result from parameterized complexity.

Theorem 2.4.8 (see, e.g., [Cyg+15, Thm. 13.28])**.** Set Cover *is* W[2]*-complete when parameterized by solution size.*

2.5 ROBUST COMBINATORIAL OPTIMIZATION

In practice the input data for optimization problems is often not known precisely. One reason is that the data is based on prediction of future prices, demands or requirements. A different source for uncertainty are measurement errors. Robust optimization is one approach to dealing with these issues.

In this section we first explain the basic idea behind robust optimization using the example of linear programming and present some classical results. Then in the next two subsections we will describe two of the fundamental approaches to robust combinatorial optimization. For more details and historic development of robust optimization in general we refer the reader to the comprehensive survey by Bertsimas, Brown and Caramanis [BBC11].

To the best of the author's knowledge, the investigation of uncertain linear programs was initiated by Soyster in the 70s. In [Soy73] the author considered an LP with column-wise uncertainty, i.e., the j-th column of the constraint matrix is part of a convex set $\mathfrak{S}_j \subseteq \mathbb{R}^n$. Each vector in $\mathfrak{S}_j$ defines a possible scenario. A robust solution to an uncertain LP is a solution that is feasible for each possible scenario. Soyster proved that under mild assumptions an optimal robust solution to uncertain linear programs of this type can be found by solving a related LP in the original space.

Ben-Tal and Nemirovski [BTN99] studied uncertain linear programs with row-wise uncertainty. The nominal, deterministic problem considered in the article is

$$\min \{c^\top x \colon Ax \geqslant b,\ x \in \mathbb{R}^n_+\}, \tag{2}$$

where $A \in \mathbb{R}^{m \times n}$. The corresponding uncertain problem, called robust counterpart, has the form

$$\begin{aligned} \min \quad & c^\top x \\ \text{s.t.} \quad & \sum_{i=1}^{m} a_i^\top x_i \geqslant b_i \quad \text{for each } a_i \in \mathfrak{S}_i,\ i \in [m], \\ & x \geqslant 0, \end{aligned} \tag{3}$$

where $\mathfrak{S}_i$ are again closed convex sets, e.g., polyhedra. We can rewrite (3) in a more compact way

$$\min \{c^\top x \colon Ax \geqslant b,\ x \in \mathbb{R}^n_+,\ A \in \mathfrak{S}\}, \tag{4}$$

where $\mathfrak{S} := \mathfrak{S}_1 \times \cdots \times \mathfrak{S}_m$. A matrix $A \in \mathfrak{S}$ represents one realization of the data and is again referred to as a scenario. The row-wise uncertainty model has a very natural property. If there is a compact convex set $C \subseteq \mathbb{R}^n$ such that C contains feasible solutions for each scenario $A \in \mathfrak{S}$, then the robust counterpart (4) is infeasible if and only if there exists an $A \in \mathfrak{S}$ such that $\{x \in \mathbb{R}^n_+ \colon Ax \geqslant b\}$ is empty [BTN99,

Prop. 2.1]. If the robust problem (4) is feasible, then its robust optimal value coincides with

$$\sup_{A \in \mathfrak{S}} \{c^\top x \colon Ax \geqslant b,\ x \in \mathbb{R}^n_+\},$$

i.e., a robust optimal solution corresponds to a solution of a worst-case scenario. The robust counterpart (3) can be reformulated as a convex problem and is tractable if there is a an efficient separation algorithm for $\mathfrak{S}$ [BTN99, Sec. 2.3]. It is worth to remark that this model can also incorporate uncertainty in the cost function c and the right hand side vector b. In both cases the uncertainty can be integrated into the constraints.

The solutions to (3) can be very conservative in terms of the objective value if compared to the nominal problem (2). One countermeasure is to use uncertainty sets that control the deviation from a nominal scenario, thus reducing the impact of "extreme" scenarios on optimal robust solutions. Examples for scenario sets with this properties is the ellipsoidal uncertainty or Γ-uncertainty by Bertsimas and Sim [BS04]. Of course the choice of the uncertainty set class can not be made completely independent of the application in mind.

2.5.1 Cost Robustness

A classical approach to robust combinatorial optimization is to assume that only the cost function is subject to uncertainty and to seek a minimum-cost worst-case solution. We summarize the robustness paradigms sharing this characteristic under the name cost robustness. The basic model is the so-called min-max model popularized by Kouvelis and Yu [KY97] that we illustrate by the robust counterpart of the classical assignment problem. The input for the robust problem consists of a graph G and an uncertainty set $\mathfrak{S} \subseteq \mathbb{R}^{E(G)}$. The set $\mathfrak{S}$ defines all possible cost functions c and each cost function defines a scenario. The optimizer's task is to find a perfect matching in G minimizing the worst-case cost over all scenarios in $\mathfrak{S}$, i.e., the task is to solve the following problem

$$\begin{aligned} \min \quad & \max_{c \in \mathfrak{S}} c(X) \\ \text{s.t.} \quad & X \text{ is a perfect matching in } G. \end{aligned} \tag{MM-AP}$$

A modification of the min-max model is the introduction of the regret function. Given a scenario c, the regret function measures the deviation of the objective value of a selected solution to the cost of an optimal solution $c(\mathrm{OPT}_c)$ for the realized scenario c. In the min-max-regret model, the task is to find a solution minimizing the maximum regret over all scenarios. The regret version of (MM-AP) is

$$\begin{aligned} \min \quad & \max_{c \in \mathfrak{S}} c(X) - c(\mathrm{OPT}_c) \\ \text{s.t.} \quad & X \text{ is a perfect matching in } G. \end{aligned} \tag{MMR-AP}$$

As for uncertain linear programs, the choice of the uncertainty set affects the tractability of the resulting robust problem and the conservatism level of the corresponding robust optimal solutions. Discrete scenario sets $\mathcal{S}$ usually lead to NP-hard robust problems, where some convex uncertainty sets $\mathcal{S}$ (e.g., if $\mathcal{S}$ is a product of intervals) lead to tractable robust problems. Bertsimas' and Sim's Γ-uncertainty addresses both issues. In this approach a reference cost function $\bar{c}$ is given and the deviation from $\bar{c}$ is controlled by a parameter Γ, resulting in a convex uncertainty set $\mathcal{S}$. This uncertainty model, if applied to a tractable 0-1-discrete optimization problem Π, results in a tractable robust counterpart of Π [BS03]. To solve the robust problem, we need to solve at most $n+1$ instances of the nominal problem Π, where n is the number of variables.

Various different cost-robustness concepts were proposed in the literature. One widely-used paradigm is given by two-stage robust optimization problems, where the decision maker has to select a partial solution in the first stage and then augments this partial solution in the second stage when the emerged scenario has been revealed. Examples for two-stage robustness concepts are recoverable robustness [Lie+09], adjustable robustness [BT+04], K-adaptability [BC10] and its special case min-max-min robustness [BK17a].

A survey on min-max and min-max-regret versions of several combinatorial optimization problems is given by Aissi, Bazgan and Vanderpooten [ABV09]. Kasperski and Zieliński [KZ16] summarized more recent results on the classical min-max model and also provided an exposition of different two-stage models. In [BK17b] Buchheim and Kurtz survey different results with focus on the uncertainty sets and their influence on the tractability of the corresponding robust combinatorial problems.

Results for cost-robust variants of matching problems from the literature will be presented in Section 2.6.1.

2.5.2 Redundancy-Based Robustness

Redundancy-based robustness is a fundamentally different concept compared to cost-robustness. The costs are (usually) certain and the uncertainty affects the constraints of the optimization problem under consideration. Because of that, robust solutions here are not necessarily feasible to the underlying nominal problem as it is the case for cost-robust problems. For a graph problem this typically means that the structure of the graph changes in different scenarios. An exemplary representative of a redundancy-based robust problem is the following. Given a graph G, the optimizer seeks a connected subgraph of G that is fault-tolerant against deletion of up to k edges from the graph. In other words the optimizer has to solve the minimum $(k+1)$-edge connected spanning subgraph problem ($(k+1)$-ECSS), which is

a famous problem in network design (see Cheriyan, Sebő and Szigeti [CSS01], and Gabow et al. [Gab+09]).

Several robust models described in the literature fit into the category of redundancy-based robustness. Dhamdhere et al. [Dha+05] proposed a two-stage model named demand-robustness. We illustrate this concept using SET COVER as an example. A demand-robust SCP instance is defined by a ground set $[k]$, a collection $\mathcal{S} \subseteq 2^{[k]}$ and a set of m scenarios, where each scenario is specified by a subset $U_i \subseteq [k]$. Those are the elements that must be covered in i-th scenario. In the first stage the optimizer has to select a set of subsets from $\mathcal{S}$ anticipatorily, without knowing which scenario is going to be realized. This information is revealed in the second stage by presenting U_i for some $i \in [m]$. In the second-stage the costs, i.e., the cardinality of the sets in $\mathcal{S}$, are inflated by a parameter $\sigma > 1$. The problem is summarized as follows

$$\begin{aligned} \min \quad & |\mathcal{C}_0| + \sigma|\mathcal{C}_i| & & \\ \text{s.t.} \quad & \mathcal{C}_0 \cup \mathcal{C}_i \text{ covers } U_i & & \text{for each } i \in [m], \\ & \mathcal{C}_0 \cup \mathcal{C}_i \subseteq \mathcal{S} & & \text{for each } i \in [m]. \end{aligned} \tag{DR-SCP}$$

Note that depending on the realized scenario U_i the sets selected from $\mathcal{S}$ in one of the two stages may be of no use. In the general version of this model, each scenario also has a different cost function. Thus, demand robustness can be seen as a hybrid robustness concept. In [Dha+05] the authors provided approximation algorithms for several classical optimization problems including minimum cut, shortest paths, Steiner trees, node cover and uncapacitated facility location. A variant of this concept, where the scenarios are given implicitly was applied to covering problems in [Fei+07].

A new approach to redundancy-based robustness is bulk robustness, a concept introduced recently by Adjiashvili, Stiller and Zenklusen [ASZ15]. As for demand robustness, the uncertainty set is given as a list of possible failure scenarios. But unlike the latter model, bulk robustness is a single-stage model. We describe the concept next. Let Π be a combinatorial optimization problem, where the optimizer has to select a subset of resources R minimizing the cost with respect to a function $c \in \mathbb{R}_+^R$. We call Π the nominal problem. Given a scenario list $\mathcal{G} \subseteq 2^R$ the robust counterpart of Π is defined as

$$\begin{aligned} \min \quad & c(X) & & \\ \text{s.t.} \quad & \forall F \in \mathcal{G}\colon X \setminus F \text{ contains a feasible solution to } \Pi, & & \\ & X \subseteq R. & & \end{aligned} \tag{BR-Π}$$

We call an element in R, which is part of some scenario F vulnerable. Observe that there is no obligation to use the resources in X once a particular scenario has emerged. The uncertainty can be uniformly distributed, i.e., $\mathcal{G} = \{F \subseteq R\colon |F| = k\}$ resulting in problems similar

to the aforementioned $(k+1)$-ECSS. But this model is more general and allows to incorporate settings where the underlying structure R is only partially vulnerable. The robust assignment problems from Chapter 3 and 4 fall into this category. In [ASZ15] the authors applied the concept of bulk-robustness to the problem of finding an s-t-path and the minimum matroid basis problem.

Various combinatorial problems following a redundancy-based robustness concept can be found in literature: facility location problems (Jain and Vazirani [JV00], Swamy and Shmoys [SS03], and Chechik and Peleg [CP10]), network design problems (Jain [Jai01], Chekuri et al. [Che+02], and Hajiaghayi, Immorlica and Mirrokni [HIM03]), spanner problems (Chechik et al. [Che+09a], and Dinitz and Krauthgamer [DK11]), flow problems (Bertsimas, Nasrabadi and Stiller [BNS13], and Matuschke, McCormick and Oriolo [MMO17]) as well as many others.

2.6 RELATED WORK

In this section we survey results from works on uncertain matchings within the field of robust optimization and related problems from graph theory.

2.6.1 Robust Matching Problems

Kouvelis and Yu [KY97] showed that the min-max assignment problem (MM-AP), p. 23, is NP-hard even if the uncertainty set $\mathcal{S}$ consists only of two scenarios. Deineko and Woeginger [DW06] showed that (MM-AP) with a fixed number of discrete scenarios is equivalent to the exact perfect matching problem, a problem not known to be strongly NP-hard at that point. The latter problem asks whether a graph has a perfect matching of a given weight. Later Zhu, Luo and Miao [ZLM08] proved that the exact matching problem is strongly NP-hard. If the number of scenarios $|\mathcal{S}|$ is constant, then (MM-AP) can be interpreted as a multi-objective problem. For this class of problems Grandoni et al. [Gra+14] presented an $(1+\varepsilon)$-approximation algorithm. For the case, where the number of discrete scenarios is unbounded, Aissi, Bazgan and Vanderpooten [ABV05] showed that (MM-AP) is strongly NP-hard. For the same setting the authors provided a $|\mathcal{S}|$-approximation in [ABV09]. Kasperski and Zieliński [KZ09] proved a lower appoximation bound of $\log^{1-\varepsilon}|\mathcal{S}|$, for any $\varepsilon > 0$ unless $\mathsf{NP} \subseteq \mathsf{DTIME}(n^{\mathrm{poly}\log n})$. All above statements also hold for the min-max regret assignment problem defined in (MMR-AP).

Kasperski, Kurpisz and Zieliński [KKZ14] studied a variant of the two-stage robust perfect matching problem under the name rent-recoverable assignment. In this problem, we are given a graph G, a set of scenarios $\mathcal{S}$ and two numbers $\alpha \in (0,1)$ and $\beta \geqslant 0$. Each scenario

$s \in \mathfrak{S}$ defines a cost function $c_s \in \mathbb{R}^{E(G)}$. We denote the set of all perfect matchings in G by $\mathcal{X}(G)$. In the first stage a perfect matching $X \in \mathcal{X}(G)$ is chosen for rent. In the second stage the optimizer has to decide whether he implements X or switches to a different solution $Y \in \mathcal{X}(G)$. The costs in the first stage, i.e., the rent costs, are defined as $c_s^1(X) = \alpha c_s(X)$ for a scenario $s \in \mathfrak{S}$. The implementation costs in the second stage are defined as $c_s^2(X) := \min_{Y \in \mathcal{X}(G)}\{(1-\alpha)c_s(Y) + (\alpha+\beta)c_s(Y \setminus X)\}$. The task is to minimize the worst-case costs, which leads to the following problem

$$\min_{X \in \mathcal{X}(G)} \max_{s \in \mathfrak{S}} c_s^1(X) + c_s^2(X). \tag{RR-AP}$$

In [KKZ14] the authors proved that (RR-AP) is NP-hard with discrete scenario set $\mathfrak{S}$ and cannot be approximated to within $\log^{1-\varepsilon}|\mathfrak{S}|$ for any $\varepsilon > 0$ unless $\mathsf{NP} \subseteq \mathsf{DTIME}(n^{\mathrm{poly}\log n})$.

[Kut+08] studied a version of the assignment problem defined on a family of graphs that fall into the category of demand-robust problems. Given a bipartite graph $G = (U \dot\cup W, E)$ and a collection $\mathcal{C}$ of subsets of U (i.e., constraints on the node set U) the task is to find a maximum-weight set of independent edges in G such that each subset of the collection $\mathcal{C}$ is covered as best as possible.

Laroche et al. [Lar+14] investigated the Nurse Rostering Problem, a problem arising in health care, which is closely related to the feasibility version of V-RAP. The authors provided a measure of a roster's sensitivity concerning the absence of staff members. We will come back to this paper in Section 4.2 in more detail.

Hassin and Rubinstein [HR02] introduced the following notion of an α-robust matching. A perfect matching M in a weighted graph is α-robust (for $\alpha \in (0,1]$), if for every $p \leqslant |M|$, the p heaviest edges of the matching have a total weight at least α times the weight of a maximum-weight matching of size p. In [HR02] the authors proved that the complete graph K_n contains a $\frac{1}{\sqrt{2}}$-robust matching and that the latter bound is tight in general. Additionally, the authors provide a polynomial-time algorithm to find such a matching. Building upon these results, Fujita, Kobayashi and Makino [FKM10] proved that the problem to decide whether a graph has an α-robust matching with $\alpha \in (\frac{1}{\sqrt{2}}, 1)$ is NP-complete, and extend the original algorithm to the matroid intersection problem. Matuschke, Skutella and Soto [MSS14] improved the bound of Hassin and Rubinstein to $\frac{1}{\ln 4}$ when instead of a deterministic matching M, a probability distribution on the set of matchings in the graph is specified.

Arulselvan et al. [Aru+16] analyzed the following modification of the assignment problem. Consider an edge-weighted bipartite graph $G = (U \dot\cup W, E)$ with lower and upper quotas $l, u \in \mathbb{Z}_+^W$ on the node set W. The goal is to find a maximum-weight edge set M such that each node in U is incident with at most one edge in M while all nodes in W must either respect the bounds given by the quota functions or

are not to be used at all. This problem is the generalization of the bipartite many-to-one matching with a dynamically adjustable node set W. The main results provide a classification of several variants of the problem in terms of complexity and an approximation algorithm for the general case.

Katriel et al. [KKMU08] studied a two-stage stochastic optimization problem on an edge-weighted bipartite graph $G = (U \dot\cup W, E)$, that is of slight resemblance to RAP. The first stage is certain. In the second stage, uncertainty comes into play in two variants: either the edge costs are uncertain, or some of the nodes from W are deactivated. For both variants, the goal is to select edges in the first stage and in the second stage in such a way that their union is as cheap as possible and contains a perfect matching in G. The main results include the derivation of lower bounds on approximation guarantees as well as provision of approximation algorithms for the different variants of the presented stochastic problems. Additionally the authors provide a randomized approximation algorithm for the robust version of their stochastic optimization problem with uncertain edge costs. The algorithm returns, with high probability, a solution that contains a matching covering at least $(1-\beta)|U|$ nodes in every scenario for any $\beta \in (0,1)$. All three problems considered in [KKMU08] are different from those studied in this thesis. E-RAP seeks to find a superset of a perfect matching while some edges may fail but the edge costs are certain. In V-RAP the task is, knowing the cost, to select a subset of nodes in W such that the induced subgraph has a U-perfect matching while nodes in W are subject to uncertainty.

Dourado et al. [Dou+15] studied the question of existence of a robust recoverable matching. That is, given a graph G, a set F of edges, and two integers r and s, does the graph G have a perfect matching M such that, for every choice F' of r edges from F, the graph $G - F'$ contains a perfect matching M' and $|M \triangle M'| \leqslant s$? Such a matching M is called r-robust and s-recoverable. The authors proved hardness of several related problems and present some tractable cases. We will study an augmentation problem related to robust recoverable matchings in Chapter 5.

2.6.2 Interdiction Problems

An interdiction problem is a two-player game on a graph. In the first step an interdictor destroys some parts of the graph according to some set of rules. In a second step the optimizer selects a solution to the underlying optimization problem in the remaining graph.

Hung, Hsu and Sung [HHS93] considered the so-called most vital edges in the context of weighted bipartite matchings. In an edge-weighted graph G, an edge $e \in E(G)$ is called most vital if its removal from G minimizes the objective value of a maximum-weight match-

ing in $G - \{e\}$. The authors provided an $O(n^3)$ algorithm to find most vital edges.

Zenklusen [Zen10] considered the following matching interdiction problem. The input is a graph $G = (V, E)$ with edge-weights w, an interdiction cost function c on the edge or the node set and an interdiction budget B. Then the interdictor has to find a subset R of edges or nodes that solve the following problem

$$\min\{w(M) \colon M \text{ is a maximum matching in } G - R,\ c(R) \leqslant B\}.$$

Several NP-hardness results are presented in the paper: for graph consisting of isolated edges as well as for general graphs with unit edge weights and unit interdiction costs. For bipartite graphs the edge interdiction problem remains NP-hard for unit weights and costs, which was already shown in [Zen+09]. However, the node interdiction problem with the same costs setting is proven to be tractable. Furthermore the author provided a constant-factor approximations for both versions of the problem in general graphs with unit interdiction costs. Additionally a pseudo-polynomial algorithm for graphs with bounded treewidth and the edge interdiction setting is presented. The paper is concluded by describing how to turn the latter algorithm into an FPTAS using a scaling and rounding technique.

Haney et al. [Han+17] proposed a symmetric interdiction approach that restricts both, the interdictor and the optimizer, to feasible solutions to the underlying optimization problem. The authors applied the model to the matching problem yielding the interdiction problem

$$\min\{\nu(G - M) \colon M \text{ is a matching in } G\},$$

where G is a given (non-bipartite) graph. Their results include a general approximation algorithm if the underlying optimization problem is a packing problem and a specialized one for the matching setting which has an approximation guarantee of 1.5. Furthermore they provided APX-hardness proof for the matching interdiction and an polynomial algorithm based on polyhedral techniques if the interdictor is allowed to select his matching randomly.

Kamalian and Mkrtchyan [KM08] investigated a similar decision problem. Given a bipartite graph G and an integer k, the question is whether there is a maximum matching M in G such that $\nu(G - M) \leqslant k$ (or $\geqslant k$). The difference to the work of Haney et al. is that in [Han+17] the matching M can be of arbitrary size. The authors proved NP-completeness of this problem using a reduction from Max-2-SAT.

2.6.3 Matching Preclusion

Plesník [Ple72] proved that for any integer $r > 0$, an $(r-1)$-edge-connected and r-regular graph G remains perfectly matchable after

removing $r-1$ arbitrary edges from G. Thus, this result provides a sufficient condition for a (non-bipartite) graph with a r-factor to have a robust assignment in case $r-1$ edges can fail simultaneously.

Brigham et al. [Bri+05] studied the minimum number of edges to be removed from a graph G to arrive at a graph without a perfect matching. This quantity is called matching preclusion number $mp(G)$. The authors determined $mp(G)$ for hypercubes, Petersen graphs and complete graphs. We will come back to $mp(G)$ in Section 3.2 in the context of feasibility testing for E-RAP. For further results and generalizations of $mp(G)$ see the works of Cheng et al. [Che+09b], Park and Ihm [PI11] and references therein.

The connection between a graph's matching number $\nu(G)$ and node deletion was investigated by Favaron [Fav96]. The author characterized the class of so-called k-factor-critical graphs, i.e., graphs on n nodes such that every subgraph on $n-k$ nodes is perfectly matchable. Those graphs can not be bipartite, hence those insights can not be applied here. Aldred et al. [AAL07] studied the conditions under which grid graphs and k-fold product graphs remain perfectly matchable after non-trivial node deletions. For more recent results in this line of research we refer to an article by Lou and Yu [LY04].

2.6.4 Graphs with Extendable Matchings

In Section 2.3.2 we introduced matching-covered graphs which are a special case of n-extendable graphs. A graph is called n-extendable if any matching of size n is contained in a perfect matching. Thus, matching-covered graphs form the class of 1-extendable graphs. Several generalizations of n-extendable graphs are presented in the literature.

Liu and Yu [LY93] introduced the notion of (m,n)-extendability. In an (m,n)-extendable graph G, for every choice of a matching M of size m and n nodes that are not incident with any of the edges in M, the graph G has a perfect matching $\bar{M}$ containing M and $\bar{M}$ does not contain any edge connecting two of the selected n nodes. Using this terminology an n-extendable graph is $(n,0)$-extendable. The authors provided different properties of (m,n)-extendable graphs.

Porteous and Aldred [PA96] suggested a different extendability property. A graph maintains the $E(m,n)$-property if, for any pair (M,N) of disjoint matchings with $|M| = m$ and $|N| = n$, the graph has a perfect matching that contains N and does not contain any element of M. The authors deduce several properties of graphs satisfying $E(m,n)$-property.

Wang, Yuan and Zhou [WYZ09] examined the so-called k-edge-deletable IM-extendable graphs that are characterized as follows. After removing any set F of k edges from a graph G, every induced

matching M in $G - F$ (i.e., a matching satisfying $E(V(M)) = M$) is contained in a perfect matching of $G - F$.

2.6.5 Augmentation Problems

A general graph augmentation problem asks the following question: How many edges have to be added to a graph to satisfy a particular property? A typical representative is the Two-Edge-Connectivity Augmentation Problem (2-ECAP). Given an edge-weighted graph $G = (V, E)$ and a set of edges $E_0 \subseteq E$, 2-ECAP asks to find a minimum-cost subset of edges $E' \subseteq E$ such that the graph $G' = (V, E_0 \cup E')$ is 2-edge-connected [GJ79, ND18]. In other words, unlike $G_0 = (V, E_0)$, the graph G' is fault-tolerant against single edge failures. Thus, from robust point of view, the initial graph G_0 is a 0-robust solution with respect to the requirement of connectivity and the resulting graph G' is a 1-robust solution. Eswaran and Tarjan [ET76] proved that the problem is NP-hard if the weights are either 1 or 2 via a reduction from Hamiltonian Cycle. The authors also provide a polynomial-time algorithm for graphs with unit weights. A 2-approximation for the general case was given by Frederickson and JáJá [FJ81].

A famous special case of 2-ECAP is the so-called Weighted Tree Augmentation Problem (WTAP) where the initial graph (V, E_0) is a tree, i.e., an undirected, connected and acyclic graph. WTAP is NP-hard even if the input tree has constant diameter [FJ81]. Until recently the best known algorithm for general WTAP was the aforementioned 2-approximation from [FJ81]. Then, for the case, where the edge weights are bounded by a constant, Adjiashvili [Adj17] presented a new LP-based approximation with performance guarantee of roughly $1.96 + \varepsilon$. Later his result was improved to $1.5 + \varepsilon$ by Fiorini et al. [Fio+17] essentially closing the gap to the best known bound of 1.5 for the unit-cost WTAP by Kortsarz and Nutov [KN16].

To the best of the author's knowledge, there are no works dealing with augmenting robust matchings using any of the models described in Section 2.5.

Şeref, Ahuja and Orlin [ŞAO09] studied network problems within the following framework: Given a feasible solution to an optimization problem, the task is to find an incremental change of this solution resulting in the best possible improvement of the objective value. Several problems including spanning tree, max flow and shortest path were studied in [ŞAO09]. The incremental variant of the assignment problem was proved to be a special case of the exact matching problem which is strongly NP-hard (see p. 26) and solvable in random pseudopolynomial time.

2.6.6 **Miscellaneous**

Sha and Steiglitz [SS93] provided a distributed reconfiguration algorithm for bipartite matching with node failures that is motivated by an application in very-large-scale integration (VLSI). The setting is the following: In a bipartite graph $G = (U \dot\cup W, E)$ with a maximum matching M, the task is to react to removal of k nodes from M. The algorithm finds a new maximum matching (if possible) in reconfiguration time $O(k \cdot \min\{|U|, |W|\})$. In the process, a matching of size at least $|M| - k$ is maintained.

Darmann et al. [Dar+11] studied the hardness of the Maximum Matching Problem (among other problems) with the additional structure of conflict (forcing) graphs. A conflict (forcing) graph describe pairs of edges such that at most (at least) one of the two can be part of a feasible solution. The problem is shown to be NP-hard even if the conflict (forcing) graph has only isolated edges. Öncan, Zhang and Punnen [OZP13] use these ideas to present further complexity results and heuristics for the Perfect Matching Problem with conflict constraints.

3

ROBUST ASSIGNMENTS WITH VULNERABLE EDGES

This chapter deals with the variant of Robust Assignment capturing uncertainty in the edge structure of the underlying bipartite graph. We call this variant Edge-Robust Assignment or E-RAP for short.

We repeat the setting here briefly. Given a balanced bipartite graph $G = (U \dot\cup W, E)$, where we refer to the nodes in U as jobs and to those in W as machines. An edge $e \in E$ encodes the property that a particular job can be performed on a specific machine. In E-RAP we seek to find a subset of edges such that all jobs can be performed simultaneously, i.e., the selected edges contain a perfect matching in G. Additionally the edge set E is subject to uncertainty, e.g., an edge can fail because equipment or additional information needed to execute a job is not available. All expected incidents are modeled via failure scenarios, each describing a subset of edges in E that may fail at the same time. We call the edges contained in a scenario vulnerable. When a scenario emerges, then the corresponding subset of edges is removed from the graph. A robust solution has to be feasible in every scenario. Before giving a formal description in Section 3.1 we present an outline of this chapter next.

OUTLINE

In Section 3.1 we formally introduce Edge-Robust Assignment and prove some basic properties. The most important one is the strong connection between feasible solutions to E-RAP and matching-covered graphs which will be exploited repeatedly in this chapter to design approximation algorithms. In Section 3.2 we will discuss whether a given instance can be tested for feasibility efficiently. We will also show that in the case where the uncertainty set is given implicitly by specifying the number of edges that can fail simultaneously, deciding feasibility is an NP-hard problem. In Section 3.3 we study E-RAP with constant number of vulnerable edges and show that E-RAP is already NP-hard with only two vulnerable edges. Our main result of this chapter, hardness of approximation of E-RAP with single edge failures, will be presented in Section 3.4. The key ingredient is an S-reduction from Set Cover implying that E-RAP is not only

NP-hard to solve exactly but also NP-hard to approximate with an approximation ratio of sublogarithmic order. Subsequently we focus on scenario sets with single edge failures for the rest of the chapter. The major algorithmic result of this thesis, a randomized approximation algorithm for E-RAP, is presented in Section 3.5. In expectation, this algorithm has an approximation guarantee of logarithmic order, hence it is asymptotically tight. The algorithm relies on properties of matching-covered graphs as well as the compact polyhedral description of the perfect matching polytope. Thereafter we address the minimum-cardinality version of E-RAP, which we call Min-Card Edge-Robust Assignment or Card-E-RAP for short. In Section 3.6 we derive APX-hardness of Card-E-RAP, which implies that there is no PTAS, unless $P = NP$. We conclude this chapter presenting an $O(1)$-approximation algorithm for Card-E-RAP in Section 3.7. The algorithm uses ear-decompositions of matching-covered graphs to construct approximate solutions to Card-E-RAP. Combined, the two last-named results imply that Card-E-RAP is APX-complete.

3.1 FORMAL DESCRIPTION AND BASIC PROPERTIES

Unless stated otherwise, every graph $G = (U \dot\cup W, E)$ considered in this chapter is bipartite, simple and balanced, i.e., $|U| = |W|$. To avoid trivially infeasible cases we assume that G has no isolated nodes. The nominal problem considered in this chapter is Min-Cost Perfect Matching (MCPMP).

In E-RAP, a bulk-robust counterpart of MCPMP, the edge set E is subject to uncertainty that is modeled via a list of edge subsets $\mathfrak{S} \subseteq 2^E$. Each $F \in \mathfrak{S}$ describes a failure scenario that leads to the removal of all edges in F from the graph G. A robust assignment in G is defined as a set of edges $X \subseteq E$ such that X contains a perfect matching for every scenario. E-RAP is formalized as Problem 3.1.1.

Problem 3.1.1 (E-RAP).

Edge-Robust Assignment	
Instance:	$\langle G, \mathfrak{S}, c\rangle$, where $G = (U \dot\cup W, E)$ is a balanced bipartite graph, $\mathfrak{S} \subseteq 2^E$ a failure scenario set and $c \in \mathbb{R}^E_+$ a cost function.
Solution:	A $\mathfrak{S}$-robust assignment X in G, i.e., $X \subseteq E$ such that for each $F \in \mathfrak{S}$ the subset $X \setminus F$ contains a perfect matching of G.
Task:	Find X minimizing $c(X)$ or decide that G has no $\mathfrak{S}$-robust assignment.

The minimum-cardinality version, i.e., where $c = \mathbb{1}$ is denoted by Min-Card Edge-Robust Assignment or Card-E-RAP for short. If $\mathfrak{S}$ consists of singletons, then we just write $\mathfrak{S} \subseteq E$. An example of a

Card-E-RAP instance and an optimal solution is illustrated in Fig. 4.

We make the following distinction for uncertainty sets. The scenario sets $\mathfrak{S}$ of the first type are given explicitly as a list of subsets of E. The second type is described implicitly by an integer k as $\mathfrak{S}_k := [E]^k = \{F \subseteq E : |F| = k\}$. We call the corresponding instances k-uniform. In the latter case an instance is then usually given as $\langle G, k, c\rangle$. Consequently we call the associated feasible solutions $\mathfrak{S}_k$-robust or just k-robust. In this sense, a perfect matching is a 0-robust solution. In the case $k = 1$ we just write $\mathfrak{S} = E$ and refer to the instances as uniform.

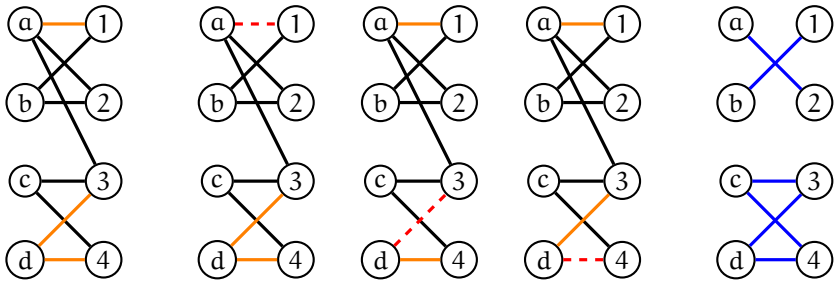

Figure 4: From left to right: a Card-E-RAP instance with $\mathfrak{S} = \{\{a, 1\}, \{d, 3\}, \{d, 4\}\}$ (orange); three scenario graphs induced by the removal of a vulnerable edge (red, dashed); an optimal solution (blue).

The restriction to balanced bipartite graphs is not limiting in the case of E-RAP with arbitrary cost functions as is shown in the next proposition.

Proposition 3.1.2. *An unbalanced* E-RAP *instance can be efficiently transformed to an equivalent balanced* E-RAP *instance while preserving the cost.*

Proof. Consider an unbalanced E-RAP instance $\langle G, \mathfrak{S}, c\rangle$, where $G := (U \dot\cup W, E)$ with $|U| < |W|$ and $c \in \mathbb{R}_+^E$. According to our motivation we want to match all jobs, i.e., nodes in U. For this reason, an instance with $|U| > |W|$ is infeasible.

To convert G into a balanced bipartite graph a set D of dummy job nodes of cardinality $|D| = |W| - |U|$ is introduced. Each $d \in D$ is connected with every machine node $w \in W$ and we denote the corresponding edges by E_D. We define a balanced graph G' by setting $W' := W$, $U' := U \cup D$ and $E' := E \cup E_D$. The new cost vector $c' \in \mathbb{R}_+^{E'}$ coincide with c on the original edge set E. The new edges have cost zero, i.e., $c'_e = 0$ for each $e \in E_D$. Ultimately, the set of failure scenarios is unaffected, i.e., $\mathfrak{S}' := \mathfrak{S}$. Evidently, this transformation can be performed in polynomial time and results in a balanced E-RAP instance $\langle G', \mathfrak{S}', c'\rangle$.

Now observe that whenever X is an assignment in G we can extend a U-perfect matching $M \subseteq X$ to a perfect matching in G' by adding suited edges from E_D. Conversely, because $U \cap V(E_D) = \emptyset$,

any perfect matching in G' contains a U-perfect matching using original edges only. Hence for an assignment X' in G', $X' \cap E$ is an assignment in G. As the new edges have zero cost, it follows that $c(X) = c'(X')$. ■

This procedure can not be applied to MIN-CARD EDGE-ROBUST ASSIGNMENT in the way presented above, because it is not approximation preserving in that case.

The next proposition establishes the connection between feasible solutions to E-RAP and matching-covered graphs. A matching-covered graph, is a graph with the property that each of its edges is contained in a perfect matching (see Section 2.3.2).

Proposition 3.1.3. *Let $\mathcal{I} = \langle G, \mathfrak{S}, c \rangle$ be an* E-RAP *instance with $\mathfrak{S} \subseteq E(G)$. Then, $X \subseteq E(G)$ is an inclusion-wise minimal feasible solution to $\mathcal{I}$ if and only if the induced subgraph $G[X]$ spans G, is inclusion-wise minimal with the properties of being matching-covered and that isolated edges in $G[X]$ are not contained in $\mathfrak{S}$.*

Proof. Let $X \subseteq E(G)$ be an inclusion-wise minimal feasible solution to $\mathcal{I}$, and let $e \in X$ be an edge. If e is dispensable, i.e., it does not appear in any perfect matching of G, then $X \setminus \{e\}$ remains feasible to $\mathcal{I}$. Thus, due to inclusion-wise minimality of X the induced subgraph $G[X]$ is matching-covered. Now assume e is vulnerable and is isolated in $G[X]$, then $X \setminus \{f\}$ cannot contain a perfect matching in G. This contradicts the feasibility of X. Lastly, assume that there is a spanning subgraph $G[X'] \subsetneq G[X]$ induced by $X' \subsetneq X$ and $G[X']$ is matching-covered and that does not share any of its isolated edges with $\mathfrak{S}$. Consider an arbitrary vulnerable edge $f \in X' \cap \mathfrak{S}$. As f is not isolated in $G[X']$, there is an $e' \in X'$ adjacent to f. Since $G[X']$ spans G and is matching-covered, there is a perfect matching M in G with $e' \in M$ and $f \notin M$. However, this implies that X' is feasible to $\mathcal{I}$ as well contradicting the inclusion-wise minimality of X.

Conversely, let X be a subset of $E(G)$ such that its induced subgraph $G[X]$ spans G and is inclusion-wise minimal with respect to both stated properties. Showing that X is feasible to $\mathcal{I}$ is similar to the proof of the feasibility of X' above. Assume now that there is an $\bar{X} \subseteq X$, $\bar{X} \neq X$, feasible to $\mathcal{I}$. This implies that the induced subgraph $G[\bar{X}]$ spans G and a vulnerable edge $f \in \bar{X} \cap \mathfrak{S}$ cannot be isolated in $G[\bar{X}]$. Moreover, $G[\bar{X}]$ can be assumed to be matching-covered. Otherwise each dispensable edge $e \in \bar{X}$ can be removed from $\bar{X}$ decreasing its cardinality. Hence, $\bar{X}$ is a proper subset of X which contradicts the minimality of X concluding the proof. ■

Consequently (inclusion-wise minimal) matching-covered graphs are natural candidates for solutions to E-RAP. Recap that dispensable

edges can be identified using the digraph $D(G, M)$ defined in Definition 2.3.7. A necessary condition for a matching-covered graph G on more than four nodes to be inclusion-wise minimal is the absence of 4-cycles [LP86, Thm. 4.2.2]. This condition can be verified efficiently. The full characterization is given by the following theorem.

Theorem 3.1.4 ([LP86, Thm. 4.2.16])**.** *Let G be a bipartite matching-covered graph. Then G is inclusion-wise minimal matching-covered if and only if no nice cycle in G has a chord.*

Although nice cycles can be recognized using the digraph $D(G, M)$ (see Definition 2.3.7) efficiently, a matching-covered graph can have an exponential number of nice cycles.

Concluding this section, we present a class of Card-E-RAP instances with a good characterization of optimal solutions.

Proposition 3.1.5. *Let $\langle G = (U \dot\cup W, E), [E]^{k-1} \rangle$ be a* Card-E-RAP *instance. If the graph G admits a k-factor, then $X \subseteq E$ is optimal if and only if X is a k-factor.*

Proof. "if" part:
Let $X \subseteq E$ be a k-factor in G. First we show the feasibility of X using Hall's Theorem 2.3.1. Consider any $F \in \mathfrak{S}$, we have to show that X contains a perfect matching in $G - F$. To do so we apply Hall's Theorem to the graph $H := G[X] - F$. Consider a subset $T \subseteq U$, we have

$$|N_H(T)| \geqslant \left\lceil \frac{k|T| - |F|}{k} \right\rceil = \left\lceil |T| - \frac{k-1}{k} \right\rceil = |T|,$$

hence X is a k-robust assignment.

Now assume, there is a feasible solution $X' \neq X$ with $|X'| < |X| = \frac{k}{2}|V(G)|$. Then, there is a node $v \in V(G)$ with $|\delta(v) \cap X'| \leqslant k-1$. Choosing a failure scenario $F' \in \mathfrak{S}$ that is a superset of $\delta(v) \cap X'$ contradicts the feasibility of X' because v is isolated in $G - F'$.

"only if" part:
Let OPT be an optimal solution to the E-RAP instance at hand. By the arguments presented above we know that $|\mathrm{OPT} \cap \delta(v)| \geqslant k$ holds for each $v \in V(G)$. Now assume there is a node v' with $|\mathrm{OPT} \cap \delta(v')| > k$ implying $|\mathrm{OPT}| > \frac{k}{2}|V(G)|$. This contradicts the optimality of OPT because G has a k-factor. ■

Finding k-factors in bipartite graphs is a tractable problem. We can either use combinatorial algorithms or polyhedral methods as described in Section 2.3.

Plesník [Ple72] showed that for a non-bipartite graph G a k-factor X is feasible for $(k-1)$-uniform E-RAP if the induced subgraph $G[X]$ is $(k-1)$-edge-connected (see p. 29). An example that this additional requirement is necessary is presented in Appendix B.

3.2 DECIDING FEASIBILITY

In this section we discuss how to decide feasibility of a given E-RAP instance $\mathcal{I} = \langle G = (U \dot\cup W), \mathfrak{S}, c\rangle$. Note that the cost function c does not influence the feasibility.

Observe that E-RAP is a monotonic problem: Every superset of a feasible solution is feasible itself. Thus, the instance $\mathcal{I}$ is feasible if and only if the entire edge set E is a feasible solution. Furthermore, note that as a decision problem, the question whether $\mathcal{I}$ at hand is feasible is contained in the complexity class coNP. If $\mathcal{I}$ is infeasible, then there is a failure set $F \subseteq E$ of size k as a short certificate.

There are two easy cases. If $\mathfrak{S}$ is given explicitly as a list of vulnerable edge sets then we can check if for every scenario $F \in \mathfrak{S}$ the graph $G - F$ has a perfect matching using any efficient matching algorithm. Same holds if $\mathfrak{S} = [E]^{\ell}$, where ℓ is a constant. Hence in both cases testing feasibility can be performed in time polynomially bounded by the size of the input.

Hence the interesting case appears if the scenario set $\mathfrak{S}$ is specified implicitly by an integer k, i.e., the instance is given as $\langle G, k, c\rangle$. Then the naive approach described above does not yield an algorithm polynomial in k, because we have to compute $\binom{|E|}{k} = O(|E|^k)$ matchings. The naive approach yields just an XP-algorithm with k as the parameter. In the remainder of this section we focus on k-uniform instances where the scenario set $\mathfrak{S} = [E]^k$ is given implicitly by providing the integer k as input.

Brigham et al. [Bri+05] investigated when the removal of edges leads to a graph that is not perfectly matchable. For this the authors introduced the matching preclusion number. For a graph G, it is defined as

$$\mathrm{mp}(G) := \min\{|F| \colon F \subseteq E(G),\ G - F \text{ has no perfect matching}\}.$$

The matching preclusion number measures a graph's edge deletion resilience with respect to property of being perfectly matchable. The definition of $\mathrm{mp}(G)$ naturally leads the following decision problem.

MATCHING PRECLUSION NUMBER (MPNP)
Instance: $\langle G, s\rangle$, where G is a graph and $s \in \mathbb{Z}_+$.
Question: Is $\mathrm{mp}(G) \leqslant s$?

The decision problem MPNP is known to be NP-complete for bipartite graphs [Lac+12, Thm. 6] (see [Dou+15, Thm. 2] for a different proof). We use this problem to derive hardness for the question of feasibility of E-RAP.

Proposition 3.2.1. *Deciding feasibility of a given* k*-uniform* E-RAP *instance* $\langle G, k, c\rangle$ *is* NP*-hard.*

Proof. We provide a Turing reduction from MATCHING PRECLUSION NUMBER. Let $\langle G, s\rangle$ be an instance of MPNP. Then, with any choice of $c \in \mathbb{R}_+^{E(G)}$, the tuple $\langle G, [E(G)]^s, c\rangle$ is a feasible E-RAP instance if and only if $mp(G) > s$. Thus we can decide whether $\langle G, s\rangle$ is a YES-instance of MPNP using a feasibility oracle for k-uniform E-RAP. ■

Note that Proposition 3.2.1 rules out the existence of approximation algorithms in the k-uniform setting with k as part of the input.

3.3 CARD-E-RAP WITH TWO VULNERABLE EDGES

In this section we study Card-E-RAP with a constant number of vulnerable edges. If only one edge is uncertain, we can just remove it from the graph. Thus the simplest non-trivial variant is Card-E-RAP has two singleton scenarios. In Theorem 3.3.3 we prove that even for this very restricted setting Card-E-RAP is already NP-hard.

In the remainder of this section we consider instances of Card-E-RAP of the form $\langle G = (U \dot\cup W, E), \{f_1, f_2\}\rangle$, where f_1 and f_2 are distinct edges in G. If the graph $G - \{f_1, f_2\}$ is perfectly matchable, then any perfect matching not using f_1 and f_2 is optimal. Hence, while investigating hardness, we neglect the instances where the vulnerable edges are not part of an optimal solution. This means that we assume both edges, f_1 and f_2, to be part of every feasible solution. We can verify if an instance $\mathcal{I}$ is feasible by computing a maximum matching M_i in $G - \{f_i\}$, $i \in \{1, 2\}$. The union $X := M_1 \cup M_2$ is a feasible solution to $\mathcal{I}$ and gives us a 2-approximation. Moreover, the set X contains a cycle C including both f_1 and f_2 and a perfect matching on the nodes not covered by C. Borrowing a term from theory of matching-covered graphs, we call such cycles nice. This observation is formalized next.

Observation 3.3.1. *Let $\langle G, \{f_1, f_2\}\rangle$ be an instance of* Card-E-RAP *such that $G - \{f_1, f_2\}$ has no perfect matching. Then every inclusion-wise minimal solution X has the form $C \cup M$, where C is a cycle containing f_1 and f_2 and M a perfect matching in $G - V(C)$. If C spans G, then M is empty.*

For the complexity analysis of Card-E-RAP with two singleton scenarios we use a reduction from the following optimization problem.

Problem 3.3.2 (SNPP)**.**

SHORTEST NICE PATH	
Instance:	$\langle G, s, t\rangle$, where $G = (U \dot\cup W, E)$ is a balanced bipartite graph, $s \in U$ and $t \in W$.
Solution:	A nice s-t-path P in G, i.e., an s-t-path such that all nodes not in $V(P)$ are perfectly matchable in G.
Task:	Find P minimizing length(P) or decide that no such path exists.

In other words in SNPP we seek to partition the graph into an s-t-path and a subgraph with a perfect matching. This problem is NP-hard (see Theorem 3.3.6), the proof is postponed to the end of this section. We are now ready to prove the main result of this section.

Theorem 3.3.3. MIN-CARD EDGE-ROBUST ASSIGNMENT *is* NP*-hard with two singleton scenarios.*

Proof. Let $\mathcal{I} = \langle G, s, t\rangle$ be an instance of SNPP, i.e., $G = (U \dot\cup W, E)$ is a balanced bipartite graph with $s \in U$ and $t \in W$. We first introduce two new nodes x, y, where x is added to W and y to U. We then add new edges $f_1 := \{s, x\}$, $f_2 := \{x, y\}$ and $f_3 := \{y, t\}$ to G (see Fig. 5). The construction yields a new balanced bipartite graph $G' = (V', E')$

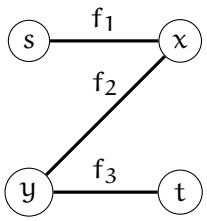

Figure 5: Reduction from Theorem 3.3.3: the new nodes and edges in G'.

with $V' := (U \cup \{y\}) \dot\cup (W \cup \{x\})$. Next, we define the set $\mathfrak{S}$ of failure scenarios as $\{f_1, f_2\}$. Then, $\mathcal{I}' := \langle G', \mathfrak{S}\rangle$ is an instance of Card-E-RAP, which can be obtained in polynomial time.

Note that for any feasible s-t-path P for $\mathcal{I}$ there exists a perfect matching M in $G[(U \dot\cup W) \setminus V(P)]$. As the path P connects nodes from different sides of the bipartition, P is of odd length. Now we interpret P as a path in G'. Then, $C := P + \{f_1, f_2, f_3\}$ is a cycle of even length in G'. By Observation 3.3.1, the union $X = M \cup C$ is a feasible solution to the Card-E-RAP instance $\mathcal{I}'$.

Since the node x is only incident with the vulnerable edges f_1 and f_2, every feasible solution to $\mathcal{I}'$ has to contain a cycle covering f_1 and f_2. Thus, a nice s-t-path in G of minimal length leads to a cycle C in G' of minimal length, and vice versa.

Now we claim that a shortest nice s-t-path P for instance $\mathcal{I}$ has length L if and only if an optimal solution X to $\mathcal{I}'$ consists of a cycle C with length $L + 3$.

Let P be a shortest nice s-t-path for $\mathcal{I}$. Using the aforementioned arguments, we obtain a feasible solution $X := M \cup C$ with $\text{length}(C) = \text{length}(P) + 3$. Since P is an optimal solution, a smaller cycle containing the vulnerable edges f_1 and f_2 does not exist. Therefore, $M \cup C$ is optimal to $\mathcal{I}'$.

Conversely, consider now an optimal solution X' to $\mathcal{I}'$. Since x and y are incident only with edges f_1, f_2 and f_3, the solution X' contains a cycle C' that covers the nodes s, t, x, y. Moreover, the remaining nodes $V(G') \setminus V(C')$ are perfectly matched, i.e., the cycle C' is nice in

G'. Because X' is optimal, the cycle C' is a shortest nice cycle of this type. By removing the edges f_i, $i = 1, 2, 3$, from C' we get a path P from s to t which is feasible to $\mathcal{I}$, i.e., P is a nice path in G. Evidently, $\text{length}(P) = \text{length}(C') - 3$.

The NP-hardness of Shortest Nice Path stated in Theorem 3.3.6 concludes the proof. ■

A natural parameter choice for a parameterized algorithm for E-RAP is the number of scenarios $|\mathfrak{S}|$. Unfortunately the question for an FPT-algorithm in $|\mathfrak{S}|$ is unlikely to have an affirmative answer.

Corollary 3.3.4. *Unless* P = NP, Edge-Robust Assignment *is not fixed-parameter tractable when parameterized by the number of failure scenarios.*

Proof. Assume there is an FPT-algorithm for E-RAP with running time $f(k) \cdot n^c$, where k is the number of scenarios, n the size of the input and c some constant. Then for E-RAP with two scenarios we have $k = 2$ and we can solve the problem in $O(n^c)$ contradicting the NP-hardness established by Theorem 3.3.3. ■

In fact Theorem 3.3.3 even rules out the existence of XP-algorithms for E-RAP when parameterized by the size of the scenario set $\mathfrak{S}$ unless P = NP.

In the remainder of this section we prove the NP-hardness of Shortest Nice Path via a reduction from a restricted variant of the following decision problem.

Problem 3.3.5 (PAFPP).

Path Avoiding Forbidden Pairs	
Instance:	$\langle D, s, t, \mathcal{FP}\rangle$, where $D = (V, A)$ is a digraph, $\mathcal{FP} \subseteq [V]^2$ set of forbidden pairs and s, t two nodes in V.
Question:	Is there a directed s-t-path P in D such that for each pair $(a, b) \in \mathcal{FP}$ at most one of the two nodes is covered by P?

Problem PAFPP is known to be NP-complete [GJ79, GT54]. We use a bipartite version of this problem suited for our needs, which we call Path Avoiding Forbidden Pairs in Bipartite Graphs (BPAFPP). An instance of BPAFPP is a tuple $\langle H, s, t, \mathcal{FP}\rangle$, where $H = (U \cup W, E)$ is an undirected balanced bipartite graph, and s,t as well as $\mathcal{FP}$ are defined as in Problem 3.3.5. Additionally the set $\mathcal{FP}$ satisfy the following properties

(P1) $|\mathcal{FP}| = k$ is even,

(P2) for each $(a_i, b_i) \in \mathcal{FP}$: either $a_i, b_i \in U$ or $a_i, b_i \in W$,

(P3) $\left|\{(a_i, b_i) \in \mathcal{FP}\colon a_i, b_i \in U\}\right| = \left|\{(a_i, b_i) \in \mathcal{FP}\colon a_i, b_i \in W\}\right|$,

(P4) $s \in U$ and $t \in W$.

The hardness of BPAFPP is proven in Appendix A. We are now ready to show the following result.

Theorem 3.3.6. Shortest Nice Path *(Problem 3.3.2) is* NP*-hard.*

Proof. We provide a Turing reduction from BPAFPP to SNPP. Let $H = (U \dot\cup W, E_H)$ be a balanced bipartite graph, $s \in U$, $t \in W$, and let $\mathcal{FP} = \{(a_1, b_1), \ldots, (a_k, b_k)\}$ be a collection of forbidden pairs, all together comprising an instance $\mathcal{I} = \langle H, s, t, \mathcal{FP}\rangle$ of BPAFPP. To obtain a corresponding SNPP instance $\mathcal{I}'$ the following five steps are performed on H and illustrated in Fig. 6.

(T1) Set $L := |U \dot\cup W| + 1$ (that is an odd number as H is balanced).

(T2) For every $i \in [k]$, introduce new nodes s^i_j, $j \in [L]$, as well as a new path $Q_i := (a_i, s^i_1, s^i_2, \ldots, s^i_L, b_i)$ connecting the two nodes of the corresponding forbidden pair (a_i, b_i) through the new nodes s^i_j, $j \in [L]$, and add both, the new nodes and the edges of Q_i, $i \in [k]$, to H.

(T3) For every node $v \in (U \dot\cup W) \setminus \{s, t\}$, introduce a new path $R_v := (v, p^v_1, p^v_2, \ldots, p^v_L)$, and add the new nodes p^v_q, $q \in [L]$ and the edges of R_v to H.

(T4) Define $K := \{p^v_L \colon v \in (U \dot\cup W) \setminus \{s, t\}\}$, and add the edge set

$$E_K := \left\{\{p^{v_1}_L, p^{v_2}_L\} \colon v_1 \in U \setminus \{s\},\ v_2 \in W \setminus \{t\}\right\}$$

to H. This yields a complete balanced bipartite subgraph $H_K := (K, E_K) \subseteq H$.

(T5) Set $s' = s$ and $t' = t$.

Let $H' := (V', E_{H'})$ be the graph after performing steps T1 – T5 on the input graph H. Note that every newly introduced edge in H' is incident with at least one node in $V' \setminus V(H)$. The bipartiteness of H' holds because the parameter L that controls the length of the new paths Q_i and R_v is odd and due to the fact that, for each forbidden pair (a_i, b_i), both a_i and b_i belong to the same node partition of H (Property P2). By Property P3 of BPAFPP, the number of forbidden pairs contained in each bipartition is the same implying H' is also balanced. Moreover, H' contains H as a subgraph, and a bipartition $U' \dot\cup W'$ of H' can be chosen such that $U \subseteq U'$ and $W \subseteq W'$, i.e., $s' \in U'$ and $t' \in W'$. Hence $\mathcal{I}' := \langle H', s', t'\rangle$ is an instance of SNPP constructed from $\mathcal{I}$ in polynomial time.

Note further that the internal nodes $\{s^i_1, s^i_2, \ldots, s^i_L\}$ of each path Q_i, $i \in [k]$, introduced in Step T2 are odd in number. Thus, all the

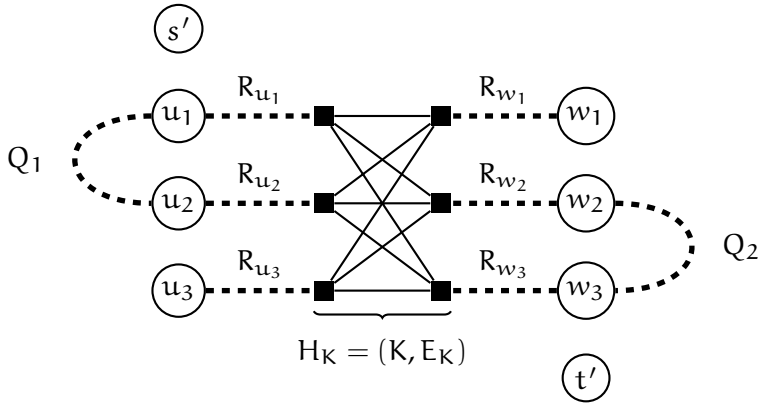

Figure 6: Reduction from Theorem 3.3.6 sketched for the BPAFPP instance $\langle H, s, t, \mathcal{FP}\rangle$ on the graph $H = (U \dot\cup W, E_H)$, where $U = \{s, u_1, u_2, u_3\}$, $W = \{w_1, w_2, w_3, t\}$ and $\mathcal{FP} = \{\{u_1, u_2\}, \{w_2, w_3\}\}$. For sake of clarity the internal nodes of the paths Q_i and R_w, as well as the edge set E_H from the original graph H are omitted.

node sets of the form $\{s_1^i, s_2^i, \ldots, s_L^i\}$ cannot be perfectly matched among themselves in H'. The same is true for the internal nodes $\{p_1^v, p_2^v, \ldots, p_L^v\}$ of any path R_v introduced in Step T3. Observe also that, for each $v \in K$, the nodes p_L^v and v are in different parts of the bipartition, which is due to the fact that L is odd.

We conclude the proof by showing the following claim and using the NP-completeness of BPAFPP.

Claim. The SNPP instance $\mathcal{I}'$ contains a nice s-t-path of length $\ell < L$ if and only if the BPAFPP instance $\mathcal{I}$ is feasible .

"only if" part:

Let P be a nice s-t-path of length $\ell < L$ in H'. We show that P is feasible to $\mathcal{I}$. Since P has at most $L - 1$ edges, it neither contains an edge from one of the paths Q_i, $i \in [k]$, nor from R_v, $v \in (U \dot\cup W) \setminus \{s, t\}$, as otherwise all $L + 1$ edges from Q_i or R_v are contained in P (see steps T2 and T3). Furthermore, since P contains no edge from any path R_v, P is also disjoint from the subgraph H_K introduced in Step T4. It follows that P is completely contained in the subgraph H of H', i.e., P is already a path in the graph H.

Next, we show that P is feasible to $\mathcal{I}$, i.e., for every $i \in [k]$, at least one of the nodes a_i, b_i is avoided by P. For this, we fix a forbidden pair (a_i, b_i). Since P is a nice s-t-path in H', there exists a matching M in H' covering all nodes in H' that are not contained in $V(P)$. As the internal nodes $s_1^i, s_2^i, \ldots, s_L^i$ of any path Q_i do not belong to P, it follows that they are covered by M. Recap moreover that L is odd. This, in particularly, means that either $\{a_i, s_1^i\} \in M$ or $\{s_L^i, b_i\} \in M$. Thus, at most one of the nodes from the forbidden pair (a_i, b_i) is incident with P.

"if" part:

Let P be a feasible solution to the BPAFPP instance $\mathcal{I}$, i.e., P is in

particular an s-t-path in H. To show that P is a nice s-t-path in H' we provide a matching M in H' that covers all nodes not in $V(P)$. For this we assume w.l.o.g. that for each forbidden pair (a_i, b_i) at most the nodes a_i are part of the path P and define the following index set

$$J := \{i \in [k] \colon a_i \in V(P)\}.$$

The matching M is constructed as follows. Start with $M := \emptyset$, and consider first the nodes on the paths Q_i, $i \in [k]$. To cover all nodes of Q_i not belonging to $V(P)$, a suitable set of independent edges (i.e., every second edge) from the path Q_i is chosen. If $i \in J$, then add the set of independent edges covering $s^i_1, \dots, s^i_L, b_i$ to M. Otherwise, i.e., if $i \in [k] \setminus J$, then add the set of alternating edges that match $a_i, s^i_1, \dots, s^i_L$ to M. Observe that M already covers either node a_i or b_i, for each forbidden pair (a_i, b_i). By properties of BPAFPP, it further follows that the number of nodes in U that are covered by M and that belong to a forbidden pair is the same as the number of nodes in W covered by M and belonging to a forbidden pair.
Second, consider all nodes in the subgraph H that do not belong to $V(P)$ and that are not yet covered by M. Let $\tilde{U} \dot\cup \tilde{W}$ denote these nodes where $\tilde{U} \subseteq U$ and $\tilde{W} \subseteq W$. Then, $|\tilde{U}| = |\tilde{W}|$ holds, which follows from the fact that $|\tilde{U}| = \frac{n}{2} - q - k = |\tilde{W}|$, where

- $\frac{n}{2} = |U| = |W|$ is the number of nodes on each side of the bipartition of H,
- $q = \frac{1}{2}|V(P)|$. More precisely, $|V(P) \cap U| = |V(P) \cap W|$ (because P connects $s \in U$ with $t \in W$, we know that $|V(P)|$ is even),
- and k is the number of pairs in $\mathcal{FP}$ (recap that M covers exactly one node of each pair by now).

To extend M to a matching in H' also covering $\tilde{U} \dot\cup \tilde{W}$, the following edges are added to M. For each $v \in \tilde{U} \dot\cup \tilde{W}$, choose the set of independent edges from path R_v that cover $v, p^v_1, p^v_2, \dots, p^v_L$.
Now observe that the only nodes in $V(H') \setminus V(P)$ still unmatched by M are the internal nodes $\{p^v_1, p^v_2, \dots, p^v_L\}$ of a path R_v that is associated with a node $v \in (U \dot\cup W) \setminus (\tilde{U} \dot\cup \tilde{W} \cup \{s, t\})$. Such a node v is either contained in $V(P)$ or v is part of a forbidden pair and is covered by an edge added to M in the first step. Most of these nodes can be covered by adding, for each $v \in (U \dot\cup W) \setminus (\tilde{U} \dot\cup \tilde{W} \cup \{s, t\})$, the edges $\{p^v_1, p^v_2\}$, $\{p^v_3, p^v_4\}, \dots, \{p^v_{L-2}, p^v_{L-1}\}$ from $E(R_v)$ to M.
This still leaves all nodes $p^v_L = \mathrm{tail}(R_v)$ with a node $v \in (U \dot\cup W) \setminus (\tilde{U} \dot\cup \tilde{W} \cup \{s, t\})$ to be unmatched by M. Let

$$K' := \left\{p^v_L \colon v \in (U \dot\cup W) \setminus (\tilde{U} \dot\cup \tilde{W} \cup \{s, t\})\right\} \subseteq K$$

be the set of all these end nodes. It remains to extend M to a matching also covering K'. Recap that K' is, as a subset of U, part of the balanced

bipartite subgraph $H_K = (K, E_K)$ constructed in Step T4. Furthermore, we have $|K' \cap U| = \frac{1}{2}|K'| = |K' \cap W|$ and the subgraph H_K contains a matching only covering nodes from K'. After adding one such matching on K' to M, the set M becomes a matching in H' that covers all nodes of H' not contained in $V(P)$. This shows that P is a nice s-t-path in H'. As P is completely contained in the subgraph H, its node set $V(P)$ can only consists of at most $|U \dot\cup W| < |U \dot\cup W| + 1 = L$ nodes, proving that the length ℓ of P is strictly less than L.
Thus the claim is proven. The statement of the theorem now follows from NP-completeness of BPAFPP. ■

3.4 COMPLEXITY OF E-RAP

In this section we prove that EDGE-ROBUST ASSIGNMENT is as hard as SET COVER even if the failure scenarios are determined by single edges. This is the main complexity result of Chapter 3.

We first provide the description of a basic reduction from SET COVER to E-RAP and then, in Theorem 3.4.3, show that it is an S-reduction.

Lemma 3.4.1. *There is a basic reduction* (f, g) *from* SET COVER *to* EDGE-ROBUST ASSIGNMENT *with* $\mathfrak{S} \subseteq E$.

Proof. We prove the four properties in Definition 2.2.3 step by step. Let $\mathcal{I} := \langle [k], \mathcal{S} \rangle$ be an arbitrary feasible instance of SET COVER and $\ell := |\mathcal{S}|$.

(B1): We start the construction of the E-RAP instance $\mathcal{I}' := f(\mathcal{I}) = \langle G, \mathfrak{S}, c \rangle$ with the definition of the graph G. The graph is obtained by performing four transformation steps described next. An example of an E-RAP instance obtained this way is illustrated in Fig. 7.

(T1) For each $i \in [k]$ we introduce a new node u_i and we define $U_{[k]} := \{u_i : i \in [k]\}$. For each $S_j \in \mathcal{S}$ we introduce a node w_{S_j} and set $W_{\mathcal{S}} := \{w_{S_j} : S_j \in \mathcal{S}\}$. Furthermore, the edge set $E_{SC} := \big\{\{u_i, w_{S_j}\} : s \in S_j, j \in [\ell]\big\}$ is added to G. These edges and nodes encode the structure of the SET COVER instance $\mathcal{I}$.

(T2) For each $i \in [k]$, a copy w_i of the node u_i is introduced and we define $W_{[k]} := \{w_i : i \in [k]\}$. Furthermore, the edge set $E_{[k]} := \big\{\{u_i, w_i\} : i \in [k]\big\}$ is added to G. These edges are declared as vulnerable to ensure that every robust assignment in the graph induces a feasible cover for $\mathcal{I}$.

(T3) For each $S \in \mathcal{S}$, three copies u_S, $\bar u_S$ and $\bar w_S$ of the node w_S are introduced. We define the sets $U_{\mathcal{S}}$, $\bar U_{\mathcal{S}}$ and $\bar W_{\mathcal{S}}$ correspondingly. Furthermore the edge set $E_{\mathcal{S}} := \big\{\{u_S, w_S\} : S \in \mathcal{S}\big\}$ is introduced. We use the edges in $E_{\mathcal{S}}$ to indicate the SCP solution within the robust assignment. Finally two matchings on the node sets $W_{\mathcal{S}}$

and U_S as well as $\bar{U}_S$ and $\bar{W}_S$ are added to G. We denote all of these 2ℓ edges by E_C. Edges in E_C are in some sense complementary, they do not represent any structure but establish the feasibility of the E-RAP instance.

(T4) For each edge $\{u_i, w_{S_j}\}$ in E_{SC}, $i \in [k]$, $j \in [\ell]$, we introduce a twin edge $\{\bar{u}_{S_j}, w_i\}$. These edges comprise the edge set $\bar{E}_{SC}$.

Note that the node set $\bar{U}_S$ and its incident edges are merely needed to ensure that the resulting graph is balanced. Applying the four steps above yields a bipartite graph $G = (U \dot\cup W, E)$, where $U := U_{[k]} \cup U_S \cup \bar{U}_S$, $W := W_{[k]} \cup W_S \cup \bar{W}_S$ and $E := E_{[k]} \cup E_{SC} \cup E_C \cup E_S \cup \bar{E}_{SC}$. We finish the construction of $\mathcal{I}'$ by defining the scenario set $\mathfrak{S} := E_{[k]}$ and the weights $c \in \mathbb{R}^E_+$, where

$$c_e := \begin{cases} 1 & \text{if } e \in E_S, \\ 0 & \text{if } e \in E \setminus E_S. \end{cases} \tag{C}$$

Observe that the presented construction leads to a well-defined function $f\colon \mathcal{I}_{\text{SCP}} \to \mathcal{I}_{\text{E-RAP}}$.

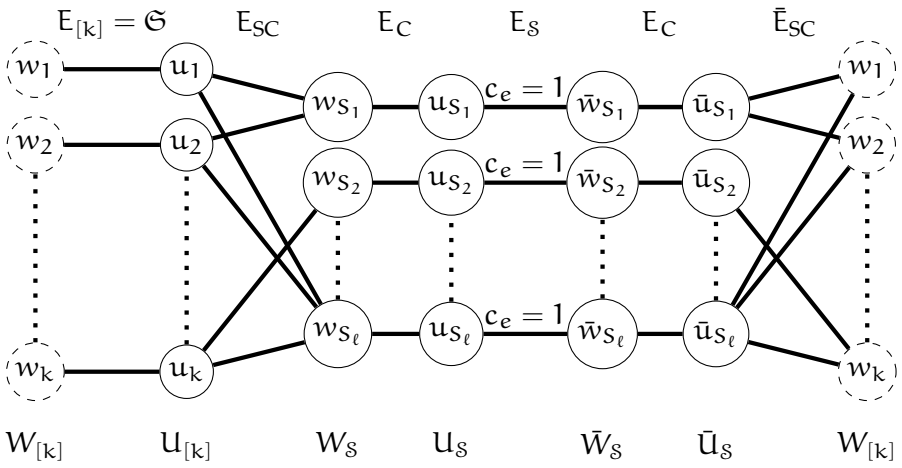

Figure 7: An E-RAP instance $\langle G, \mathfrak{S}, c\rangle$ constructed by the basic reduction (f, g) from an SCP instance $\langle [k], \{S_1, \dots, S_\ell\}\rangle$ (see Lemma 3.4.1). For the sake of clarity the node set $W_{[k]}$ appears twice.

(B2): Let $\mathcal{C} \subseteq \mathcal{S}$ be a cover for $\mathcal{I}$. We claim that

$$X_{\mathcal{C}} := (E \setminus E_S) \cup \big\{\{u_S, \bar{w}_S\} \colon S \in \mathcal{C}\big\}$$

is a robust assignment for $\mathcal{I}'$. Recap that $X_{\mathcal{C}} \subseteq E$ is feasible to $\mathcal{I}'$ if and only if $X_{\mathcal{C}} \setminus \{f_i\}$ contains a perfect matching of G, for every $f_i = \{w_i, u_i\} \in \mathfrak{S} = E_{[k]}$. Let $f_i \in \mathfrak{S}$. Observe that $M := E_{[k]} \cup E_C$ is a perfect matching in G with $M \subseteq X_{\mathcal{C}}$. The matching M contains the vulnerable edge f_i but it can be adjusted appropriately as we see next. As $\mathcal{C}$ is a cover for $\mathcal{I}$, there is a set $S_j \in \mathcal{C}$ such

that $i \in S_j$ implying $\{u_{S_j}, \bar{w}_{S_j}\} \in X_{\mathcal{C}}$. Now consider the cycle $C_i = (w_i, u_i, w_{S_j}, u_{S_j}, \bar{w}_{S_j}, \bar{u}_{S_j}, w_i)$ containing the vulnerable edge f_i. Note that $E(C_i) \subseteq X_{\mathcal{C}}$ and C_i is an M-alternating cycle. Hence $M \triangle C_i$ is a perfect matching in $G[X_{\mathcal{C}} \setminus \{f_i\}]$.

(B3): Let $X' \in \texttt{sol}(\mathcal{I}')$ be an arbitrary robust assignment. We claim that

$$\mathcal{C}_{X'} := \{S \in \mathcal{S} \colon \{u_S, \bar{w}_S\} \in X'\}$$

is a cover for $\mathcal{I}$. Let $i \in [k]$. We have to show that i is covered by some set $S \in \mathcal{C}_{X'}$. Consider scenario $f_i \in \mathfrak{S}$. As X' is a robust assignment, there is a perfect matching $M \subseteq X'$ in G with $f_i \notin M$. Since $f_i \notin M$, node u_i must be matched to a node in $W_{\mathcal{S}}$ via an edge in E_{SC}, i.e., $\{u_i, v_S\} \in M$, for some $S \in \mathcal{S}$. By definition of E_{SC} we know that $i \in S$. Because the node w_S is already matched, its neighbor u_S is saturated using an edge in $E_{\mathcal{S}}$, implying $S \in \mathcal{C}_{X'}$. Hence $C_{X'}$ is a cover and we define $g(\mathcal{I}, X') := C_{X'}$.

(B4): By construction of G, we have that $|V(G)| = 2k + 4\ell$ and $|E(G)| = k + 3\ell + 2\sum_{j=1}^{\ell} |S_j| \leqslant k + 3\ell + 2k\ell$, i.e., the input size of G is polynomially bounded by the size of $\mathcal{I}$. Because $\mathfrak{S} = E_{[k]}$ and $c \in \{0,1\}^E$, the function f can be computed in time polynomial in size$(\mathcal{I})$. Since the function g has simply to go through every edge in X', g is polynomial-time computable as well. ■

Lemma 3.4.1 can be adjusted to obtain a reduction to uniform E-RAP in the following way. Recap that the matching $E_{[k]} \cup E_C$ is unaffected by failure of edges in $E_{SC} \cup E_{\mathcal{S}} \cup \bar{E}_{SC}$. Thus, the main obstacle is that if an edge in E_C fails, then an edge in $E_{\mathcal{S}}$ has to be included into the robust assignment. But this fact destroys any approximation preserving properties of the reduction because then feasible assignments always correspond to trivial covers. We can eliminate this issue by performing an additional transformation step on the graph G as constructed by the reduction from Lemma 3.4.1.

(T5) Each edge $e \in E_C$ is replaced by a cycle C_e of length six. We denote the new edges $\bigcup_{e \in E_C} E(C_e)$ by E'_C (see Fig. 8).

This procedure yields a new, slightly larger, bipartite graph that we denote by $G' := (U' \dot{\cup} W', E')$. The graph G as obtained via function f share most edges with G', i.e., $E' = (E(G) \setminus E_C) \cup E'_C$.

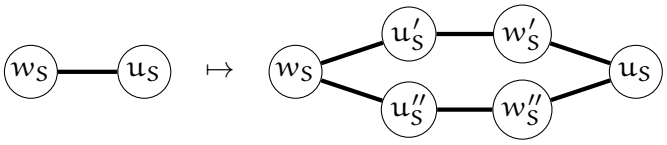

Figure 8: Replacing an edge $\{u_S, w_S\} \in E_C$ by a cycle of length six.

Now if one of the edges in C_e fails and is hence removed from G', its absence can be compensated within the cycle C_e. Hereby we maintain the property that the edges in E_S indicate the SCP solution encoded by the robust assignment. On the other hand a matching that uses e in G can use three independent edges from C_e instead, implying the feasibility of the robust assignment corresponding to $X_{\mathcal{C}}$ as defined in proof of Lemma 3.4.1. We omit further, rather tedious, details and summarize the result in the next lemma.

Lemma 3.4.2. *There is a basic reduction* (f, g) *from* Set Cover *to* Edge-Robust Assignment *with* $\mathfrak{S} = E$.

We will now show that (f, g) is an S-reduction, proving the announced hardness result.

Theorem 3.4.3. *For every* $\varepsilon > 0$, *it is* NP*-hard to approximate* Edge-Robust Assignment *with* $\mathfrak{S} = E$ *to within* $(1 - \varepsilon) \ln n$, *where* n *is the number of nodes in the underlying bipartite graph.*

Proof. We prove the claim by showing that the basic reduction (f, g) defined in Lemma 3.4.2 satisfies the two properties of an S-reduction (see Definition 2.2.5). Let $\mathcal{I}$ be a feasible instance of Set Cover and $\mathcal{I}' := f(\mathcal{I}) = \langle G, \mathfrak{S}, c \rangle$ the corresponding E-RAP instance with $\mathfrak{S} = E$.

(S1): Let X' be a feasible solution to $\mathcal{I}'$ and $\mathcal{C}_{X'} := g(\mathcal{I}, X') = \{S \in \mathcal{S} \colon \{u_S, \bar{w}_S\} \in X'\}$. Recap the definition of the cost function in (C). It follows immediately that $c(X') = c(X' \cap E_S) = |\mathcal{C}_{X'}|$.

(S2): Observe that by construction of f and g, an optimal E-RAP solution $\mathrm{OPT}(\mathcal{I}')$ is mapped to an optimal cover $\mathrm{OPT}(\mathcal{I})$ implying that both optimal objective values coincide.

Hence, E-RAP inherits all inapproximability properties of Set Cover stated in Theorem 2.4.4. ∎

Recap that S-reductions can be also used to deduce parameterized hardness with respect to the standard parameterization as well as APX-hardness. Hence using the results for (2,3)-SCP stated in Theorem 2.4.7 and parameterized hardness for general SCP stated in Theorem 2.4.8 we immediately obtain the following consequences.

Corollary 3.4.4. E-RAP *is* APX*-hard and* NP*-hard to approximate to within* $\frac{100}{99}$ *even if the maximum degree of the underlying graph is at most five. Furthermore, the standard parameterization of* E-RAP *in general graphs is* W[2]*-hard.*

3.5 O(log n)-APPROXIMATION FOR E-RAP

In this section we present a randomized approximation algorithm for E-RAP with uncertainty set $\mathfrak{S} \subseteq E$, which is the main algorithmic result of this thesis. The algorithm uses a randomized rounding technique based on a decomposition of a fractional matching. The expected approximation ratio is $O(\log n)$, where n is the number of nodes in the underlying graph. This means, the algorithm matches the lower approximation bound established in Theorem 3.4.3 up to multiplicative constants.

All instances of E-RAP in this section are feasible. Consider an instance $\mathcal{I} = \langle G = (U \dot\cup W, E), \mathfrak{S}, c\rangle$. We first discuss the case where $\mathcal{I}$ is a uniform instance, i.e., $\mathfrak{S} = E$ and later extend the algorithm to handle non-uniform instances. We denote the number of nodes and edges in G by n and m, respectively. The basic idea of the algorithm is the following. We start with a perfect matching in G and include it to an intermediate solution X. Then in each iteration we choose a scenario $f \in X \cap \mathfrak{S}$, such that $X \setminus \{f\}$ does not contain a perfect matching for G and add a perfect matching M in $G - \{f\}$ to X. Every edge $f \in \mathfrak{S}$ is considered at most once and the selected edges X induce a matching-covered graph implying that X is indeed a feasible solution to $\mathcal{I}$ (see Proposition 3.1.3). Of course, in order to obtain any bounds on the cost of X, we have to be careful while selecting the matchings for inclusion into the solution. In addition, the final algorithm will include only a part of a perfect matching.

In order to succeed with the selection process, we use an optimal fractional solution to the linear relaxation of the natural ILP formulation for E-RAP. We describe this ILP model next. For each vulnerable edge $f \in \mathfrak{S}$ we introduce a vector $y^{-f} \in \{0,1\}^E$ representing a perfect matching in the corresponding scenario graph $G - \{f\}$. Additionally a variable $x \in \{0,1\}^E$ encodes a feasible solution to E-RAP. Using these $m^2 + m$ variables, E-RAP can be modeled as an integer linear program as follows

$$
\begin{aligned}
\min \quad & c^\top x & & & \\
\text{s.t.} \quad & y^{-f} & \in & \ PM(G) & \text{for each } f \in \mathfrak{S}, \\
& y^{-f}_f & = & \ 0 & \text{for each } f \in \mathfrak{S}, \\
& x & \geqslant & \ y^{-f} & \text{for each } f \in \mathfrak{S}, \\
& y^{-f} & \in & \ \{0,1\}^E & \text{for each } f \in \mathfrak{S}, \\
& x & \in & \ \{0,1\}^E, &
\end{aligned}
\tag{E-RAP-ILP}
$$

where $PM(G)$ is the perfect matching polytope associated with G, i.e., $PM(G) = \operatorname{conv}\{\chi^M : M \text{ is a perfect matching in } G\}$. Recap that $PM(G)$ can be described by the node-edge incidence matrix of G, hence the constraints on y^{-f} are described by a TU matrix and therefore we can optimize over $PM(G)$ efficiently via the ellipsoid method.

But due to the coupling constraints $x \geqslant y^{-f}$ the constraint matrix of (E-RAP-ILP) loses this useful property. It is straightforward to verify that solutions to (E-RAP-ILP) coincide with feasible solutions to $\mathcal{I}$ and vice versa. The LP-relaxation (LP) results from relaxing all integrality constraints in (E-RAP-ILP).

We are now ready to outline the algorithm which is formalized as Algorithm 3. The algorithm begins by computing a (fractional) solution (x, y) to (LP), where y is the concatenation of the vectors $y^{-f} \in \{0,1\}^E$, $f \in \mathfrak{S}$. Then an empty edge set X is defined, which is augmented in each iteration until X is a feasible solution to $\mathcal{I}$. In Lemma 3.5.1 we prove that this is exactly the case when there are no isolated edges left in the induced subgraph $G[X]$ and $G[X]$ is matching-covered.

We now describe the augmentation process in more detail. In each iteration a vulnerable edge f is detected in Step 4 such that $X \setminus \{f\}$ contains no perfect matching. Then the algorithm randomly selects a matching in $G - \{f\}$ with sufficiently small cost using the fractional optimal solution (x, y). As a component of y, the vector y^{-f} is contained in $\mathrm{PM}(G - \{f\})$, hence y^{-f} can be represented as a convex combination of the vertices of the latter polytope. Recap that since G is bipartite the matching polytope $\mathrm{PM}(G - \{f\})$ is integral, which implies that its vertices are incidence vectors of perfect matchings. This means there exists a convex combination of y^{-f} of the following form

$$y^{-f} = \sum_{i \in [k]} \lambda_i^{-f} \chi^{M_i^{-f}},$$

where $M_1^{-f}, \ldots, M_k^{-f}$ are perfect matchings in $G - \{f\}$ and $\lambda_1^{-f}, \ldots, \lambda_k^{-f}$ positive scalars with $\sum_{i \in [k]} \lambda_i^{-f} = 1$. By Caratheodory's theorem, we can bound k, the number of vectors in the convex combination, by $m + 1$. Furthermore given y^{-f}, such a decomposition with rational scalars λ_i^{-f} can be computed in polynomial time using linear programming techniques [GLS93, Thm. 6.5.11]. The algorithm computes a convex combination of the latter type in Step 5.

The scalars λ_i^{-f} induce a probability distribution on the set of matchings $\mathcal{M}^{-f} := \{M_i^{-f} : i \in [k]\}$ and the algorithm selects a matching $\bar{M}$ from $\mathcal{M}^{-1}$ with probability λ_i^{-f} in Step 6. Subsequently the algorithm determines the subset $S \subseteq \bar{M}$ of edges connecting different components in $G[X]$ (see Fig. 9 for an illustration, note that the edge $\{d, 2\}$ is not included into the solution). After augmenting X with the edges in S, the algorithm proceeds to the next iteration. Note that both end nodes of the edge f chosen at the beginning of the iteration are covered by the edge set S, because $S \subseteq \bar{M}$ and $\bar{M}$ is a perfect matching in $G - \{f\}$.

Before proving the correctness of the algorithm in Lemma 3.5.1, we want to remark why we do not add the entire matching $\bar{M}$ selected in Step 6 to the solution in each iteration. The latter approach could

Algorithm 3: Randomized O(log n)-approximation for E-RAP

Input: A feasible E-RAP instance $\mathcal{I} = \langle G, \mathfrak{S} = E(G), c \rangle$.
Output: A robust assignment in G.

1 $(x, y) \leftarrow$ optimal solution to the relaxation (LP)
2 $X \leftarrow \emptyset$
3 **while** X is infeasible **do**
4 Select $f \in \mathfrak{S}$ such that $X \setminus \{f\}$ contains no perfect matching
5 Compute a convex combination $\sum_{i=1}^{k} \lambda_i^{-f} \chi^{M_i^{-f}}$ of y^{-f}
6 Select $\bar{M} \in \{M_i^{-f} : i \in [k]\}$ with $\mathbb{P}\left[\bar{M} = M_i^{-f}\right] = \lambda_i^{-f}$, $i \in [k]$
7 Detect edges $S \subseteq \bar{M}$ connecting distinct components in $G[X]$
8 $X \leftarrow X \cup S$
9 **end**
10 **return** X

lead to a bad performance on particular instances. In case the graph G has a node v with high degree, adding a matching for each $f \in \delta(v)$ could lead to a solution with value $\Omega(n)$ times the optimal cost.

We now prove that Algorithm 3 computes a feasible solution after a polynomial number of steps using structural results from the theory of matching-covered graphs.

Lemma 3.5.1. *Algorithm 3 returns a feasible solution X to a uniform* E-RAP *instance $\mathcal{I}$ in polynomial time. Moreover, for every set of edges $\bar{X}$ selected by the end of any iteration of the algorithm, the induced subgraph $G[\bar{X}]$ is matching-covered.*

Proof. Let $\mathcal{I}$ be a uniform instance defined on the bipartite graph $G = (U \dot{\cup} W, E)$, $X \subseteq E$ an edge set returned by Algorithm 3 and $X^i \subseteq E$ the edges selected by the algorithm in the first $i - 1$ iterations. First, we show the second property, i.e., that the subgraph $G[X^i]$ is matching-covered. Then the feasibility of X follows by Proposition 3.1.3.

We prove the second property by induction on the number of iterations. Let S^1 be the set of edges selected in Step 7 in the first iteration. Since the algorithm starts with an empty set of edges X, every node in G is a component in $G[X]$. Thus, the edge set S^1 is the entire matching $\bar{M}$ chosen randomly in Step 6 in the first iteration and $G[S^1]$ is matching-covered by definition. Thus, the claim holds for the first iteration. Since the algorithm never deletes edges that have been already chosen, we have $S^1 \subseteq X^i$, for each iteration i. We denote S^1 by M for the sake of clarity.

Now consider iteration i for a fixed $i > 1$. Let $S^i \subseteq \bar{M}$ be the set of edges selected in Step 7 in iteration i and $G^i := G[X^i \cup S^i]$ the graph induced by the edges $X^i \cup S^i$ selected by the end of iteration i. We have to show that G^i is matching-covered. By the induction

hypothesis the graph $G^{i-1} := G[X^i]$ is matching-covered. Hence we only have to show that each edge in S^i is contained in some perfect matching in G^i.

Fix $e \in S^i$. We will construct a perfect matching containing e. Let C be a cycle in G^i containing e and minimizing $|E(C) \cap \bar{M}|$. Such a cycle exists because M and $\bar{M}$ are both perfect matchings in G. In Fig. 9, for $e = \{a, 3\}$, the cycle C is $(a, 1, b, 2, c, 3)$. Let $H_1, \ldots, H_\ell$ be the components in G^{i-1} that are connected by edges in $E(C) \cap S^i$. Because the edges in S^i come from a perfect matching $\bar{M}$, each H_j shares an even number of nodes with S^i. Now assume there is an H_j, $j \in [\ell]$, such that $|V(S^i) \cap V(H_j)| \geqslant 4$. By the induction hypothesis we know that H_j is matching-covered and hence 2-node-connected. Consequently we can take a shortcut within H_j, thus decreasing the size of C and contradicting the minimality of C. This means, for all $j \in [\ell]$ we have $|V(S^i) \cap V(H_j)| = 2$. Let u_j, w_j be the two unique nodes in $V(S^i) \cap V(H_j)$, for a fixed $j \in [\ell]$. If H_j has only two nodes, namely u_j and w_j, then we define $N_j := \emptyset$. Observe that in this case u_j and w_j are incident with some edges in $S^i \cap E(C)$. Otherwise let N_j be a perfect matching in $H_j - \{u_j, w_j\}$. Such a matching exists because H_j is matching-covered by assumption (see Theorem 2.3.4). Then $L := (E(C) \cap S^i) \cup N_1 \cup \cdots \cup N_j$ is a matching in G^i covering all nodes in the components $H_1, \ldots, H_\ell$. In Fig. 9, for $e = \{a, 3\}$, the matching L is $\{\{a, 3\}, \{b, 1\}, \{c, 2\}, \{d, 4\}\}$. The matching L is not perfect, the nodes not matched by L are those contained in components of G^{i-1} that are disjoint to the cycle C. Let $V' := V(G) \setminus V(L)$. Nodes in V' can be matched using edges in M. Hence $L \cup (M \cap E(V'))$ is a perfect matching in G^i.

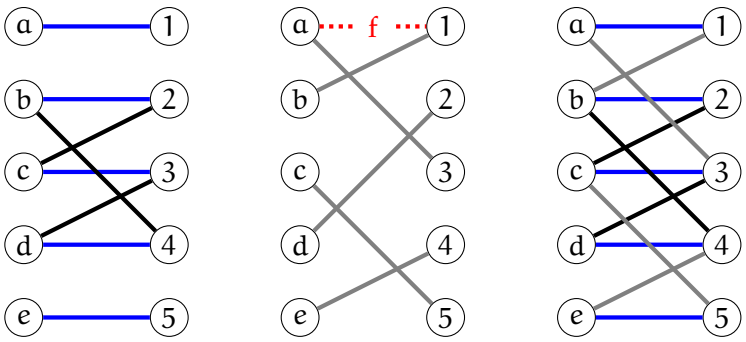

Figure 9: Transition from iteration $i-1$ to iteration i in Algorithm 3. From left to right: edges selected in the first $i-1$ iterations and the matching M selected in the first iteration (blue); a vulnerable edge f and the matching $\bar{M}$ (gray) selected in iteration i; edges contained in X by the end of iteration i.

Because $G[X^i]$ is matching-covered, every edge $f \in \mathfrak{S}$ selected at the beginning of each iteration is isolated in $G[X^i]$. By the end of the same iteration f is part of a matching-covered component of size at least four and can not chosen again. Hence the algorithm terminates after

at most $|M| = \frac{|V(G)|}{2}$ iterations. Because each step in the algorithm can be performed in time polynomial in size($\mathcal{I}$), the running time of the algorithm is polynomially bounded by size($\mathcal{I}$) as well. ■

We still have to analyze the cost of the computed solution. This is carried out with the help of a charging procedure assigning the cost of the edges selected in each iteration to some nodes in the graph.

Lemma 3.5.2. *The expected approximation guarantee of Algorithm 3 is* $O(\log n)$*, where* n *is the number of nodes in the underlying bipartite graph.*

Proof. Let $\mathcal{I} = \langle G = (U \dot\cup W, E), \mathfrak{S} = E, c\rangle$ be an instance of E-RAP with $V := U \cup W$. Let $X \subseteq E$ be the solution returned by Algorithm 3, (x, y) an optimal fractional solution to the relaxation (LP) computed in Step 1 and OPT an optimal solution to $\mathcal{I}$. For a subset $E' \subseteq E$ of edges, we denote the contribution of E' to the optimal cost of the relaxation (LP) by $c_{LP}(E') := \sum_{e\in E'} c_e x_e$.

To obtain the desired bound on the expected approximation guarantee, a scheme charging every edge in X to one of its end points is developed. We then show that the expected cost charged to a node $v \in V$ is bounded by $O(\log n)$ times the fractional cost $c_{LP}(\delta(v))$. Hence we have

$$\mathbb{E}[c(X)] \leqslant O(\log n) \sum_{v\in V} c_{LP}(\delta(v)) \leqslant O(\log n) c_{LP}(E), \tag{5}$$

where the second inequality holds due to linearity of expectation and the fact $c_{LP}(E) = \frac{1}{2}\sum_{v\in V} c_{LP}(\delta(v))$. Recap that because (LP) is a relaxation of (E-RAP-ILP) we have $c(OPT) \geqslant c^\top x = c_{LP}(E)$. Combined with (5) this proves the claim $\mathbb{E}[c(X)] \leqslant O(\log n)c(OPT)$.

Next, we describe how the cost of the selected edges is charged to the nodes of the graph. For each node v in G, this procedure formally defines a collection of edges $X_v \subseteq X$ such that $\bigcup_{v\in V} X_v = X$ holds. We first explain the construction of the edge sets X_v, $v \in V$, and then prove the required bound on $\mathbb{E}[c(X_v)]$.

The algorithm starts with $X_v = \emptyset$ for each $v \in V$. Assume we are in iteration i of the algorithm. Let $\bar X$ be the set of edges selected by the algorithm in the first $i-1$ iterations. Let $S \subseteq E \setminus \bar X$ be the set of edges chosen in Step 7 of the current iteration to augment X. At this stage, the sets X_v might already contain some edges that were added in preceding iterations. Now fix an edge $e = \{u, w\} \in S$. Recap that in order to be included into S by the algorithm, the edges in S have to connect different connected components in the subgraph $G[\bar X] = (U \dot\cup W, \bar X)$. Let C_u and C_w be the components with $u \in C_u$ and $w \in C_w$. We assume without loss of generality that $|V(C_u)| \leqslant |V(C_w)|$ holds. Then, the edge e is included in X_u. In other words, every edge added by the algorithm is charged to the end node contained in the smaller connected component, with ties broken arbitrarily. Evidently,

all edges in X are charged to some node in the course of the algorithm, such that $\bigcup_{v \in V} X_v = X$ holds when the algorithm terminates.

It remains to analyze $\mathbb{E}[c(X_v)]$ for a node $v \in V$. The bound on $\mathbb{E}[c(X)]$ then follows from linearity of expectation. To obtain the desired logarithmic bound it suffices to prove the following two properties. First, for an edge e charged to v, its expected cost is at most $c_{LP}(\delta(v))$. Second, edges can be charged to the same node v at most $\log |V|$ times, i.e., $|X_v| \leqslant \log |V|$ for every $v \in V$.

Let e be an edge selected at random in some iteration by the algorithm and charged to the node v, i.e., $e \in X_v$. The contribution of the edge e to the cost $c(X_v)$ defines a random variable Z_v^e. We show $\mathbb{E}[Z_v^e] \leqslant c_{LP}(\delta(v))$ next. Recap that any edge e selected by the algorithm was part of a perfect matching $\bar{M}$ chosen in Step 6 at random from a convex combination $y^{-f} = \sum_{i \in [k]} \lambda_i^{-f} \chi^{M_i^{-f}}$ of a fractional perfect matching y^{-f}. The scalars in the convex combinations define a distribution over the integral matchings M_i^{-f} and also induce a distribution over the edge set of the graph G. Hence, an edge e is contained in the perfect matching $\bar{M}$ with probability

$$\mathbb{P}\left[e \in \bar{M}\right] = \sum_{i \in [k] \,:\, e \in M_i^{-f}} \lambda_i^{-f} = y_e^{-f}.$$

Then, for any edge e the expected value of Z_v^e is bounded as follows

$$\begin{aligned}\mathbb{E}[Z_v^e] &= \sum_{e' \in \delta(v)} c_{e'} \cdot \mathbb{P}\left[e' \in \bar{M}\right] = \sum_{e' \in \delta(v)} c_{e'} y_{e'}^{-f} \\ &\leqslant \sum_{e' \in \delta(v)} c_{e'} x_{e'} = c_{LP}(\delta(v)),\end{aligned}$$

where the first inequality holds because $y_e^{-f} \leqslant x_e$, for all $e \in E$ by the definition of (E-RAP-ILP). This shows the first property.

To show the logarithmic bound on the number of times an edge is charged to a node v in the course of the algorithm, consider any iteration in which some edge was charged to the node v. Let C_v be the component containing v at the beginning of the iteration. Since an edge is only charged to its end node with the smaller component, and because selected edges always merge components, the size of the component containing v by the end of the iteration is at least $2|V(C_v)|$. Evidently, this duplication can happen at most $\log |V| = \log n$ times. Hence we have

$$\mathbb{E}[c(X_v)] \leqslant \log n \cdot c_{LP}(\delta(v)).$$

Then the identity $2c_{LP}(E) = \sum_{v \in V} c_{LP}(\delta(v))$ and linearity of expectation imply

$$\mathbb{E}[c(X)] \leqslant O(\log n) \sum_{v \in V} c_{LP}(\delta(v)) \leqslant O(\log n) c_{LP}(E),$$

which in combination with $c(OPT) \geqslant c_{LP}(E)$ concludes the proof. ■

We can now prove the main result of this section, which states that E-RAP with arbitrary single edge failures admits an $O(\log n)$-approximation.

Theorem 3.5.3. *Algorithm 3 is a randomized $O(\log n)$-approximation for* EDGE-ROBUST ASSIGNMENT *with $\mathfrak{S} \subseteq E$.*

Proof. In the uniform case the claim follows from Lemma 3.5.1 and 3.5.2. Thus, it remains to show how to treat non-uniform instances $\mathfrak{S} \subsetneq E$. We provide a reduction to the uniform setting.

Given a non-uniform instance $\mathcal{I} = \langle G = (U \dot\cup W, E), \mathfrak{S}, c\rangle$, we first double each non-vulnerable edge $e \in E \setminus \mathfrak{S}$ by introducing a parallel copy $\bar e$. These edges form the set $\bar E$ and we have $E' := E \cup \bar E$. This procedure specifies a graph $G' := (U \dot\cup W, E')$. The edge costs c are carried over, i.e., the cost vector $c' \in \mathbb{R}_+^{E'}$ satisfies $c'_e = c_e$, for each $e \in E$ and $c'_{\bar e} = c_e$, for each $\bar e \in \bar E$. Defining $\mathfrak{S}' := E'$ completes the construction of the new uniform instance $\mathcal{I}' := \langle G', \mathfrak{S}', c'\rangle$. Now let ALG' be a solution to $\mathcal{I}'$ returned by the algorithm. Then $ALG := ALG' \cap E$ is a solution to $\mathcal{I}$ and we have $c(ALG) \leqslant c(ALG')$. Conversely for any solution X to the non-uniform instance $\mathcal{I}$, the edge set $X' := X \cup \{\bar e \colon e \in X \setminus \mathfrak{S}\}$ is a solution to $\mathcal{I}'$ with cost $c'(X') \leqslant 2c(X)$. Hence we have $c(ALG) = O(\log n)c(OPT)$, where OPT is an optimal solution to $\mathcal{I}$. ∎

In fact the constant in the $O(\log n)$ term is at most 4. In the proof of Lemma 3.5.2, we lose a factor of 2 in (5). For non-uniform instances we obtain an additional factor of 2 by doubling of edges in the proof of Theorem 3.5.3.

We conclude the section by arguing that Algorithm 3 can be modified to return a solution to a given E-RAP instance $\mathcal{I}$ with similar approximation quality with high probability, i.e., with probability $1 - (\text{size}(\mathcal{I}))^{-K}$ for some constant $K > 0$. This can be achieved using the standard repetition technique (see, e.g., [WS11, Sec. 13.2]). Let r be the approximation ratio function associated with Algorithm 3. Due to Theorem 3.5.3 we know that for all instances $\mathcal{I}$

$$\mathbb{E}\left[r(\mathcal{I})\right] \leqslant 4 \log n.$$

Then for any $\varepsilon > 0$, Markov's inequality yields

$$\mathbb{P}\left[r(\mathcal{I}) \geqslant (1+\varepsilon) 4 \log n\right] \leqslant \frac{1}{1+\varepsilon}.$$

The new algorithm for E-RAP is denoted by $\mathcal{A}(\varepsilon, t)$ and defined as follows: We execute Algorithm 3 t times on an instance $\mathcal{I}$ and return X_t, the solution of minimum cost among the t computed solutions. The solution X_t satisfies

$$\mathbb{P}\left[\frac{c(X_t)}{c(OPT)} \geqslant (1+\varepsilon) 4 \log n\right] \leqslant \left(\frac{1}{1+\varepsilon}\right)^t.$$

This means the algorithm $\mathcal{A}(\varepsilon, t)$ has an approximation guarantee of $(1+\varepsilon)4\log n$ with probability $1-(1+\varepsilon)^{-t}$. Then, for any fixed $\varepsilon > 0$ and $t := K\log_{1+\varepsilon}(\text{size}(\mathcal{I}))$ the algorithm $\mathcal{A}(\varepsilon, t)$ is an $O(\log n)$-approximation with high probability. The running time of $\mathcal{A}(\varepsilon, t)$ is $O(\log \text{size}(\mathcal{I}))$ times the running time of Algorithm 3.

3.6 COMPLEXITY OF CARD-E-RAP

In this section we prove that MIN-CARD EDGE-ROBUST ASSIGNMENT is APX-hard. This is shown via an extension of the reduction from Section 3.4. The previous reduction is turned into an L-reduction in order to exploit the APX-hardness results for NODE COVER in cubic graphs stated in Theorem 2.4.7.

Lemma 3.6.1. *There is a basic reduction* $(\bar{f}, \bar{g})$ *from* SET COVER *to* MIN-CARD EDGE-ROBUST ASSIGNMENT *with* $\mathfrak{S} = E$.

Proof. We show the four properties of a basic reduction (see Definition 2.2.3) step by step. Let $\mathcal{I} := \langle [k], \mathcal{S}\rangle$ be an arbitrary feasible instance of SET COVER and $\ell := |\mathcal{S}|$. We adjust the basic reduction (f, g) from Lemma 3.4.2 to derive a reduction $(\bar{f}, \bar{g})$ tailored for the new context.

(B1): We start the construction of the Card-E-RAP instance $\mathcal{I}' := \bar{f}(\mathcal{I}) = \langle \bar{G}, \bar{\mathfrak{S}}\rangle$ with the graph $\bar{G}$. First we use f to obtain the graph $G' = (U' \dot\cup W', E')$.

In order to be able to control the cost of a solution later on, we need to ensure that all edges in $E_{[k]}$ are contained in any robust assignment for $\mathcal{I}'$. Subsequently, we apply the following additional transformation step.

(T6) We subdivide edges in $E_{[k]}$ into three edges, i.e., an edge $e = \{w_i, u_i\}$ is replaced by a path $P_e := (w_i, u_i', w_i', u_i)$. See Fig. 10 for an example. The new edges $\bigcup_{e\in E_{[k]}} E(P_e)$ form the set $\bar{E}_{[k]}$.

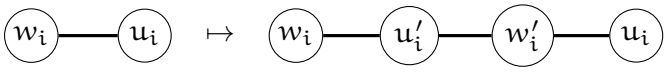

Figure 10: Replacing an edge $\{w_i, u_i\} \in E_{[k]}$ by a path of length three.

Applying Step T6 to G' yields a bipartite graph $\bar{G} = (\bar{U} \dot\cup \bar{W}, \bar{E})$, where $\bar{E} := \bar{E}_{[k]} \cup E_{SC} \cup E_C' \cup E_S \cup \bar{E}_{SC}$ and the node set $\bar{U} \dot\cup \bar{W}$ is defined accordingly. We finish the construction of $\mathcal{I}'$ by specifying the scenario set $\bar{\mathfrak{S}} := \bar{E}$.

Observe that the presented construction leads to a well-defined function $\bar{f}\colon \mathfrak{I}_{SCP} \to \mathfrak{I}_{\text{Card-E-RAP}}$.

(B2): Let $\mathcal{C}$ be a feasible cover for $\mathcal{I}$. We claim that

$$X_{\mathcal{C}} := (\bar{E} \setminus E_{\mathcal{S}}) \cup \{\{u_S, \bar{w}_S\} \colon S \in \mathcal{C}\}$$

is feasible for $\mathcal{I}'$. We follow the proof of 3.4.1 and whenever a perfect matching contains an edge in $E_{[k]}$ or E_C we replace it by two or three independent edges from $\bar{E}_{[k]}$ or E'_C.

(B3): Let $X' \in \mathtt{sol}(\mathcal{I}')$ be a robust assignment. Here no changes are necessary. The set

$$\mathcal{C}_{X'} := \{S \in \mathcal{S} \colon \{u_S, \bar{w}_S\} \in X'\}$$

is a cover for $\mathcal{I}$. We can repeat the arguments from the proof of Lemma 3.4.1 for the scenario determined by the edge $\{w'_i, u_i\}$ instead of $\{w_i, u_i\}$ (see Fig. 10). Hence $\bar{g}(\mathcal{I}, X') := C_{X'}$.

(B4): The number of edges in the graph $\bar{G}$ compared to $|E(G')|$ increased by $2k$ and the function $\bar{g}$ is not changed at all. Hence $\bar{f}, \bar{g}$ are polynomial-time computable functions.

■

In fact, the basic reduction $(\bar{f}, \bar{g})$ is an L-reduction, a fact we use to show the APX-hardness of Card-E-RAP.

Theorem 3.6.2. Card-E-RAP *with* $\mathfrak{S} = E$ *is* APX*-complete and* NP*-hard to approximate to within* $\frac{3961}{3960} \approx 1.0003$*, even in graphs with maximum degree five.*

Proof. We show the claim via a reduction from Node Cover in cubic graphs. Because 3-NCP is APX-complete (see Theorem 2.4.7) it suffices to provide an L-reduction to prove APX-hardness of Card-E-RAP. Recap that by Observation 2.4.6, 3-NCP can be restated equivalently as (2,3)-SCP. Next, we show that the basic reduction $(\bar{f}, \bar{g})$ defined in Lemma 3.6.1 is an L-reduction when restricted to (2,3)-SCP.

Let $\mathcal{I} = \langle [k], \{S_1, \ldots, S_\ell\}\rangle$ be a feasible instance of (2,3)-SCP and $\mathcal{I}' := \bar{f}(\mathcal{I}) = \langle \bar{G}, \bar{\mathfrak{S}}\rangle$ the corresponding Card-E-RAP instance. Note that the maximum degree in $\bar{G}$ is bounded by five because each $S \in \mathcal{S}$ has size three (the degree is five after applying Step T5 to the graph constructed in Lemma 3.4.1 and illustrated in Fig. 7). To simplify the analysis we introduce the notion of efficient solutions. A solution X to $\mathcal{I}$ is called efficient if it satisfies

$$|X \cap E_{SC}| = |X \cap \bar{E}_{SC}| = k. \tag{eff}$$

In other words an efficient Card-E-RAP solution X contains only the edges in $E_{SC} \cup \bar{E}_{SC}$ that are absolutely necessary for the feasibility of X. Note that any robust assignment X can be modified in polynomial time in order to satisfy (eff) and without breaking feasibility. First, for

each node u_i, $i \in [k]$, we fix a set $S_j \in \{S \in \mathcal{S} \colon \{u_S, \bar{w}_S\} \in X\}$ with $i \in S_j$. Then,

$$X_{\mathit{eff}} := \big(X \setminus (E_{SC} \cup \bar{E}_{SC})\big) \cup \big\{\{u_i, w_{S_i}\}, \{\bar{u}_{S_i}, w_i\} \colon i \in [k]\big\}$$

is feasible to $\mathcal{I}'$ and complies with (eff). Evidently, $|X_{\mathit{eff}}| \leqslant |X|$.

Let X'_{eff} be a feasible assignment for $\mathcal{I}'$ complying with (eff) and $\mathcal{C}_{X'_{\mathit{eff}}} = \bar{g}(\mathcal{I}, X'_{\mathit{eff}})$ the associated cover. Because edges in $\bar{E}_{[k]}$ and E'_C are incident with nodes of degree two and every edge can fail, these edge sets are contained entirely in any feasible assignment. Additionally we have $2k = 3\ell$ because $\mathcal{I}$ is an (2,3)-SCP instance. Hence we have

$$|X'_{\mathit{eff}}| = 5k + 12\ell + |\mathcal{C}_{X'_{\mathit{eff}}}| = 13k + |\mathcal{C}_{X'_{\mathit{eff}}}|. \tag{6}$$

We can prove the two properties of an L-reduction now.

(L1): The function $\bar{g}$ maps an optimal assignment $\mathrm{OPT}(\mathcal{I}')$ to an optimal cover $\mathrm{OPT}(\mathcal{I})$. Because we have $|S_j| = 3$, any cover of $[k]$ is of size at least $\frac{k}{3}$, i.e., $k \leqslant 3\mathtt{val}^*(\mathcal{I})$. Since any optimal solution to $\mathcal{I}'$ is efficient we can use (6) obtaining

$$\mathtt{val}^*(\mathcal{I}') = 13k + \mathtt{val}^*(\mathcal{I}) \leqslant 40\mathtt{val}^*(\mathcal{I}).$$

This means that the first parameter is $\alpha = 40$.

(L2): Let X' be an arbitrary robust assignment for $\mathcal{I}'$. We transform X' into an efficient solution X'_{eff} as described above. Recap $|X'_{\mathit{eff}}| \leqslant |X'|$. Furthermore, because the edge set E_S is not affected by the transformation we have $\mathcal{C}_{X'} = \mathcal{C}_{X'_{\mathit{eff}}}$ (see Step B3 in Lemma 3.6.1). For both absolute errors then holds:

$$\begin{aligned} |\mathcal{C}_{X'}| - \mathtt{val}^*(\mathcal{I}) &= |\mathcal{C}_{X'_{\mathit{eff}}}| - \mathtt{val}^*(\mathcal{I}) \\ &= |X'_{\mathit{eff}}| - 13k - (\mathtt{val}^*(\mathcal{I}') - 13k) \\ &= |X'_{\mathit{eff}}| - \mathtt{val}^*(\mathcal{I}') \leqslant |X'| - \mathtt{val}^*(\mathcal{I}'). \end{aligned}$$

Hence the second parameter of the L-reduction is $\beta = 1$.

Consequently, the reduction $(\bar{f}, \bar{g})$ is an L-reduction with $\alpha = 40$ and $\beta = 1$. Let r' be the approximation ratio for some E-RAP approximation and r the ratio of the approximation for 3-NCP induced by $(\bar{f}, \bar{g})$. We know that $r \leqslant (1 + \alpha\beta(r' - 1))$. Plugging the best known lower bound for 3-NCP of $\frac{100}{99}$ into the latter inequality yields a lower bound on the approximation ratio for Card-E-RAP of $\frac{3961}{3960}$. Conclusively, the membership in APX follows by Theorem 3.7.2, which we show in the next section. ■

3.7 CONSTANT-FACTOR APPROXIMATION FOR CARD-E-RAP

In this section we present a constant-factor approximation for Card-E-RAP with $\mathfrak{S} \subseteq E(G)$. The algorithm uses properties of matching-covered graphs and derives an approximate solution from an ear decomposition of the given graph. Note that ear-decomposition based approaches were successfully exploited to derive approximation algorithms for other hard combinatorial problems (e.g., 2-edge-connected spanning subgraph [CSS01] and TSP [SV14]).

Now consider an instance $\mathcal{I} = \langle G, \mathfrak{S} \rangle$ of Card-E-RAP with scenario set $\mathfrak{S} \subseteq E(G)$. We assume that $\mathcal{I}$ is feasible which can be verified efficiently (see Section 3.2). The algorithm is outlined next and formalized as Algorithm 4. In the first step, the graph is preprocessed and all edges in G that are not contained in some matching are removed. These are the so-called dispensable edges and by definition these edges cannot be part of any feasible solution anyway. Dispensable edges can be identified efficiently using DFS by Observation 2.3.8. If there are dispensable edges in G, then the resulting graph G' is disconnected (see p. 18). Hence in Step 3 the connected components H_j, $j \in [\ell]$, of G' are determined using DFS and we treat each component separately. Then in Step 5 we compute an ear decomposition of the current component H using Algorithm 1. Recap that such a decomposition exists by Theorem 2.3.6 because G' is matching-covered. Further, recall from Definition 2.3.5 that an ear decomposition consists of odd paths $P_0, \dots, P_q$ such that $H = P_0 + \cdots + P_q$. For our analysis the starting edge P_0 can be chosen arbitrarily. Note also that even after fixing the first edge an ear decomposition is not unique in general. Ears that consist of a single edge are called trivial, except for P_0. These ears do not cover any new nodes and hence are not crucial for the feasibility of the emerging solution. Thus, we ignore the trivial ears and include the edges from the remaining ears to the intermediate solution in Step 6 of the loop. Afterwards, the algorithm proceeds with the next component of G'. An illustration of a solution obtained this way is provided in Fig. 11.

Algorithm 4: An $O(1)$-approximation for Card-E-RAP

Input: A Card-E-RAP instance $\langle G, \mathfrak{S} \rangle$ with $\mathfrak{S} \subseteq E(G)$.
Output: A feasible solution X for $\langle G, \mathfrak{S} \rangle$.

1 Obtain the subgraph G' by removing dispensable ears from G
2 $X \leftarrow \emptyset$
3 Determine the connected components $H_1, \dots, H_\ell$ of G'
4 **foreach** $j \in [\ell]$ **do**
5 Compute an ear decomposition $(P_0, \dots, P_q)$ of H_j
6 $X \leftarrow X \cup E(P_0) \cup \{E(P_i) : |E(P_i)| \geqslant 3,\ i \in [q]\}$
7 **end**
8 **return** X

Note that this algorithm can be seen as a primal one. Essentially it starts with $E(G')$ which is a feasible solution because G' is matching-covered and then arrives at a solution $X \subseteq E(G')$ with smaller size by deleting trivial ears.

The correctness, the bound on the size of the computed solution as well as running time of Algorithm 4 are proven in Lemma 3.7.1. By the end of this section we explain how to extend the algorithm in order to handle non-balanced graphs. The discussion about the approximation guarantee is deferred to Theorem 3.7.2.

Lemma 3.7.1. *Let $\mathcal{I} = \langle G = (U \dot\cup W, E), \mathfrak{S}\rangle$ be a feasible* Card-E-RAP *instance with $\mathfrak{S} \subseteq E$. Algorithm 4 returns a feasible solution $X \subseteq E$ for $\mathcal{I}$ in polynomial time and $|X|$ is bounded by $3|U|$.*

Proof. Each step of the algorithm can be performed in polynomial time as discussed above. Since the number of iterations equals the number of components ℓ, which is polynomial in the input size, the algorithm terminates after a polynomial number of steps. Let G' be the subgraph obtained from G in Step 1 by removing dispensable ears and X the solution returned by the algorithm. Evidently, both, G' and $G[X]$ span G because the algorithm deletes only edges not nodes. Let H be any component of G'. Omitting trivial ears in Step 6 from the ear decomposition of H results in a valid ear decomposition for the graph induced by $X \cap E(H)$. Hence, by Proposition 3.1.3 the edge set X is a feasible solution because the graph $G[X \cap E(H)]$ is matching-covered.

It remains to bound the number of edges in X. Let H be again a component of G' and $X_H := X \cap E(H)$. Observe that H has at least 2 nodes (4 in the uniform case $\mathfrak{S} = E$) because $\mathcal{I}$ is assumed to be feasible. Let $(Q_0, \dots, Q_p)$ be the ear decomposition of $G[X_H]$ resulting from the one computed in Step 5 of the algorithm by skipping trivial ears. Note that if H is just one edge, then $p = 0$ and we have $H = P_0$. Thus the bound is valid because H contains only one job node. Now consider the size of H is at least 4, i.e., $p \geqslant 1$ and we have a substantial ear decomposition. We denote by n_j the number of job nodes in U that are internal nodes of the ear Q_j for each $j \in [p]$. Since the ears Q_j are non-trivial we have $n_j \geqslant 1$, i.e., each ear Q_j covers at least one new job node. Furthermore, we have $p \leqslant |U_H| - 1$ and $\sum_{j=1}^{p} n_j = |U_H| - 1$, where $U_H := U \cap V(H)$. Then we have

$$|X_H| \leqslant 1 + \sum_{j=1}^{p} (2n_j + 1) = 1 + 2(|U_H| - 1) + p \leqslant 3|U_H|.$$

Because X is the union of all X_H, where H is a component of G', the latter inequality yields the desired upper bound for $|X|$. ∎

We can now prove that Algorithm 4 is a constant-factor approximation for Card-E-RAP with singleton scenarios.

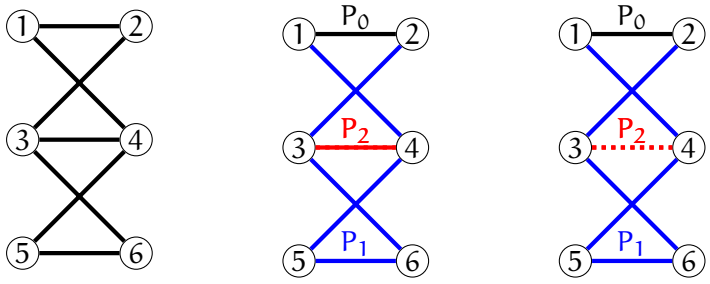

Figure 11: Illustration of Algorithm 4. From left to right: Input graph, its ear decomposition and a solution computed by the algorithm.

Theorem 3.7.2. *Algorithm 4 is a* 1.5*-approximation when applied to uniform* Card-E-RAP *instances. In general its approximation ratio is* 3*.*

Proof. Let X be a solution returned by the algorithm and OPT an optimal solution to the instance under consideration. By Lemma 3.7.1 X is feasible and its size is bounded by $3|U|$. Moreover, the algorithm terminates in polynomial time.

In the uniform case $\mathfrak{S} = E$, every node in U must have at least two incident edges in every feasible solution. Hence, $|OPT| \geqslant 2|U|$ and the approximation ratio is

$$\frac{|X|}{|OPT|} \leqslant \frac{3|U|}{2|U|} = \frac{3}{2}.$$

In the general case $\mathfrak{S} \subseteq E$ an optimal solution OPT can be much smaller, even just a perfect matching. As a result we have $|OPT| \geqslant |U|$, implying an approximation guarantee of 3. ■

The algorithm does not produce inclusion-wise minimal solutions. A computed solution is likely to contain nice cycles with chords which is by Theorem 3.1.4 equivalent to not being inclusion-wise minimal.

Remark 3.7.3. The size of the solution computed by Algorithm 4 depends on the number of trivial ears in the ear decomposition: the larger the number, the smaller the solution's size. Unfortunately finding an ear decomposition with a maximal number of trivial ears is NP-hard.

We sketch the proof here, for details see [Dah16, Thm. 5.1]. Hamiltonian Cycle is known to be NP-complete in connected bipartite graphs [Kri75]. The problem remains intractable in matching-covered bipartite graphs because dispensable ears can not be used in a Hamiltonian cycle and thus can be removed safely. Now let $(P_0, \ldots, P_q)$ be an ear decomposition of G with minimal number of trivial ears and X the edges contained in the non-trivial ears of this decomposition. Then G has a Hamiltonian cycle if and only if $|X| = |V(G)|$.

We claim that the bound 1.5 for the approximation ratio for uniform instances established in Theorem 3.7.2 is sharp. To illustrate this we provide a family of graphs, for which the algorithm may produce solutions of size arbitrary close to 1.5|OPT|.

Example 3.7.4. For $k \geqslant 3$, let G_k be a bipartite graph with node set $\{0, 1, \dots, 2k+1\}$. The edge set of G_k is defined as follows: G_k contains the paths P_i, $i \in \{2z \colon z \in [k]\}$, from 0 to 1 through nodes i and $i+1$. Additionally, G_k contains the cycles $C_j = (j, j+1, j+2, j+3, j)$, $j \in \{2z \colon z \in \{2, \dots, k-1\}\}$. Fig. 12 shows the graph G_3 on the left.

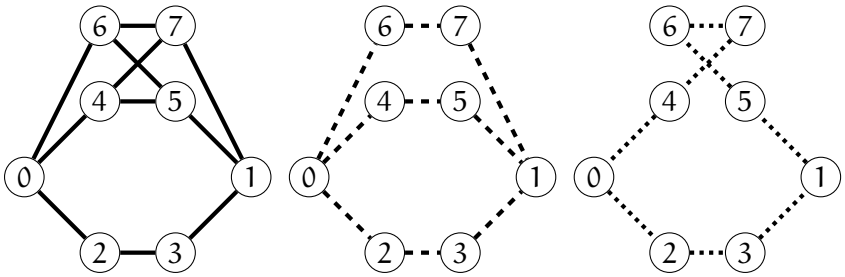

Figure 12: From left to right: Graph G_3, a worst-case solution found by Algorithm 4 and an optimal solution for $\mathfrak{S} = E$.

In the uniform case, an optimal solution OPT is a 2-factor, e.g., the Hamiltonian cycle as depicted in Fig. 12 on the right. A 2-factor has a size of $2k+2$, while a worst case solution that can be found by Algorithm 4 is given by the union of all paths P_i, $i \in \{2z \colon z \in [k]\}$ (the dashed graph in Fig. 12), since this is an ear decomposition of G. We denote the latter solution by ALG. This yields

$$\frac{|ALG|}{|OPT|} = \frac{3k}{2k+2} \xrightarrow[\infty]{k} \frac{3}{2}.$$

We further remark, that the subgraph of G_k depicted second in Fig. 12 is well-known in the literature. It illustrates the sharpness of the bound $|E(G)| \leqslant \frac{3|V(G)|-6}{2}$ on the number of edges in an inclusion-wise minimal matching-covered graph G [LP77, Thm. 2].

To conclude this section we show how to use Algorithm 4 if the underlying graph is unbalanced.

Corollary 3.7.5. *Algorithm 4 can be adapted for unbalanced* Card-E-RAP *yielding the same approximation bound.*

Proof. Consider a Card-E-RAP instance $\mathcal{I}$ defined on an unbalanced bipartite graph $G = (U \dot\cup W, E)$ with $|U| < |W|$ and scenario set $\mathfrak{S} \subseteq E$. We can transform G to a balanced bipartite graph $G' = (U' \dot\cup W, E')$ by adding a set of dummy job nodes D of size $|W| - |U|$ and connecting each of them to all nodes in W (see Proposition 3.1.2 for details).

The set of vulnerable edges $\mathfrak{S}'$ is the union of $\mathfrak{S}$ and the newly introduced edges.

Now consider $X' \subseteq E'$ the set of edges returned by Algorithm 4. Because edges in $E' \setminus E$ are only incident with dummy jobs in D, the set $X := X' \cap E$ is feasible to $\mathcal{I}$. Let Q_j be a non-trivial ear in the decomposition of G' as in Lemma 3.7.1. The ear Q_j may contain a node $d \in D$, but since the two edges incident with d are not contained in X the above analysis from the proof of Theorem 3.7.2 carries over to the new setting. The arguments in Theorem 3.7.2 remain also valid and hence Algorithm 4 has an approximation guarantee of 1.5 or 3 depending whether the instance $\mathcal{I}$ is uniform or not. ■

4

ROBUST ASSIGNMENTS WITH VULNERABLE NODES

In this chapter we study a different type of Robust Assignment capturing uncertainty in the node structure of the underlying bipartite graph. This variant is called Node-Robust Assignment or V-RAP for short.

We briefly sketch the setting first. Given a bipartite graph $G = (U \dot\cup W, E)$, we refer to the nodes in U as jobs and to those in W as machines. The edge set E is defined as follows: the edge $\{u, w\}$ is present in E if and only if the job u can be executed on machine w. The task is to find a subset of machines such that all jobs can be performed simultaneously, i.e., no pair of jobs share a machine. Additionally the set W is subject to uncertainty, e.g., a machine can become unavailable due to hardware failure or maintenance downtime. All expected incidents are modeled via failure scenarios, each describing a subset of W that may fail at the same time. When a scenario emerges, then the corresponding subset of machine nodes is removed from the bipartite graph G. A robust solution has to be feasible in every scenario. Before giving a formal description in Section 4.1 we present an outline of this chapter next.

OUTLINE

In Section 4.1 we formally introduce Node-Robust Assignment. Then in the subsequent section we describe that the feasibility of a given instance can be tested efficiently. Our main result, the hardness of approximation of V-RAP with singleton uncertainty scenarios, will be presented in Section 4.3. The key ingredient is a reduction from Set Cover that transfers all inapproximability properties from SCP to V-RAP. In Section 4.4 an approximation algorithm matching the theoretical lower bound (up to a constant additive factor) is provided. Thereafter we will address the minimum-cardinality version of V-RAP, which we call Card-V-RAP. In Section 4.5 a proof for APX-hardness is presented, implying that there is no PTAS, unless $P = NP$. Then in Section 4.6 we will refine the analysis of the previously described algorithm when restricted to Card-V-RAP and prove that it is a constant factor approximation. The two last-named results imply

the APX-completeness of Card-V-RAP. Afterwards the polynomiality of Card-V-RAP with two singleton scenarios will be shown, which distinguishes Card-V-RAP from Card-E-RAP, where two edge failures already lead to an NP-hard problem (see Section 3.3). Finally in Section 4.8 we will present a compact polyhedral description for V-RAP with uniform failure scenarios that is based on a supermodular function.

4.1 FORMAL DESCRIPTION AND BASIC PROPERTIES

Every graph $G = (U \dot\cup W, E)$ considered in this chapter is bipartite, simple and satisfies $|U| < |W|$. We call the elements in U jobs and those in W machines. To avoid trivial infeasible cases we assume that every graph G has no isolated nodes.

The deterministic problem considered in this chapter is the following node-version of MIN-COST PERFECT MATCHING. Given a bipartite graph $G = (U \dot\cup W, E)$ we want to select a set of machines X such that all jobs can be performed simultaneously on these machines. If a machine is included into a solution, then all of its incident edges can be used. In other words we are looking for a set of machines $X \subseteq W$ such that the induced subgraph $G[U \dot\cup X]$ has a U-perfect matching M, i.e., $M \subseteq E(U, X)$. We call such a set $X \subseteq W$ an assignment in G. The problem is formalized as follows.

NODE-INDUCED ASSIGNMENT (NIAP)	
Instance:	$\langle G, c\rangle$, where $G = (U \dot\cup W, E)$ is a bipartite graph with $\|U\| < \|W\|$ and $c \in \mathbb{R}^W_+$ a cost function.
Solution:	An assignment X in G, i.e., a node set $X \subseteq W$ such that the induced graph $G[U \dot\cup X]$ contains a U-perfect matching.
Task:	Find X minimizing $c(X)$ or decide that G has no assignment.

NODE-INDUCED ASSIGNMENT is a slight variation of the usual matching problem and can be described by the following ILP.

$$\begin{aligned} \min \quad & \sum_{w \in W} c_w x_w & & \\ \text{s.t.} \quad & z(\delta(w)) \leqslant x_w & & \text{for each } w \in W, \\ & z(\delta(u)) = 1 & & \text{for each } w \in U, \\ & x \in \{0,1\}^W, & & \\ & z \in \{0,1\}^E. & & \end{aligned} \tag{NIAP-ILP}$$

The convex hull P of the feasible solutions to (NIAP-ILP) is described by the following matrix $\bar{A} := \begin{pmatrix} A & -\mathbb{I}_{W\times W} \\ O_{U\times W} \end{pmatrix}$, where A is the node-edge incidence matrix of the graph G. Since G is bipartite, we have that A is totally unimodular and hence $\bar{A}$ is TU as well. Consequently P is integral and we can solve (NIAP-ILP) via its LP relaxation using the ellipsoid method (see p. 15).

In the robust version of NODE-INDUCED ASSIGNMENT, the set of machines W is subject to uncertainty that is given via failure scenarios. A robust assignment in G is a set $X \subseteq W$ such that X contains an assignment for every scenario. The formal definition is presented in Problem 4.1.1.

Problem 4.1.1 (V-RAP).

NODE-ROBUST ASSIGNMENT	
Instance:	$\langle G, \mathfrak{S}, c\rangle$, where $G = (U \dot\cup W, E)$ is a bipartite graph, $\mathfrak{S} \subseteq 2^W$ a failure scenario set and $c \in \mathbb{R}_+^W$ a cost function.
Solution:	A $\mathfrak{S}$-robust assignment X in G, i.e., $X \subseteq W$ such that for each $F \in \mathfrak{S}$ the graph $G[U \dot\cup X] - F$ contains a U-perfect matching.
Task:	Find X minimizing $c(X)$ or decide that G has no $\mathfrak{S}$-robust assignment.

The cardinality version of V-RAP, i.e., where $c = \mathbb{1}$ is called MIN-CARD NODE-ROBUST ASSIGNMENT or Card-V-RAP for short. If $\mathfrak{S}$ consists of single nodes, then we just write $\mathfrak{S} \subseteq W$.

An example of a Card-V-RAP instance and an optimal solution is illustrated in Fig. 13. We differentiate between two cases of failure scenario sets. The scenario set $\mathfrak{S}$ can be given explicitly as a list of subsets of W. The second type is described implicitly by an integer k as $\mathfrak{S}_k := [W]^k = \{F \subseteq W : |F| = k\}$. We call the corresponding V-RAP instances k-uniform. A k-uniform instance can be also given as $\langle G, k, c\rangle$. Consequently we call the associated feasible solutions k-robust. In this sense, a U-perfect matching is a 0-robust solution. In the case $k = 1$ we just write $\mathfrak{S} = W$ and refer to the instances as uniform. Evidently, a feasible k-uniform solution X satisfies $|U| + k \leqslant |X|$.

We conclude this section by characterizing inclusion-wise minimal $\mathfrak{S}_1$-robust solutions to V-RAP, which can be seen as an analog of Proposition 3.1.3 for E-RAP, where the connection between feasible solutions and matching-covered graphs is established. Consider the following generalization of Hall's Theorem

$$\forall\, T \subseteq U\colon |T| + 1 \leqslant |N_G(T)|.$$

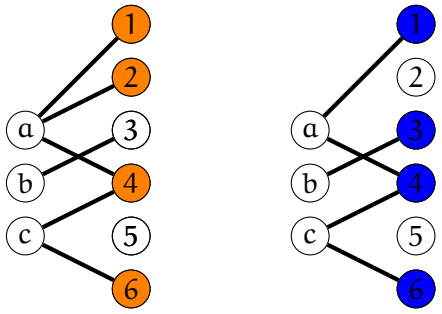

Figure 13: On the left: A non-uniform Card-V-RAP instance with four failure scenarios given by the uncertainty set $\mathfrak{S} = \{1, 2, 4, 6\}$ (orange). On the right: A $\mathfrak{S}$-robust solution (blue).

Laroche et al. [Lar+14, Thm. 2] proved that $X \subseteq W$ is feasible for V-RAP with $\mathfrak{S} = W$ if and only if the extended Hall condition holds for the graph $G[U \dot\cup X]$ (see next section for details). This fact leads immediately to the following observation.

Observation 4.1.2. *Let $\langle G = (U \dot\cup W), \mathfrak{S}\rangle$ be a uniform instance of* Card-V-RAP. *Then a set $X \subseteq W$ is inclusion-wise minimal feasible if and only if X is inclusion-wise minimal according to the following properties:*

1. *The induced graph $H := G[U \dot\cup X]$ has a U-perfect matching;*
2. *Every node in U has degree at least two in H;*
3. *Every connected component of H is odd.*

Unfortunately these properties are not directly useful for development of algorithms, hence the algorithm for V-RAP presented in Section 4.4 uses a different approach.

4.2 DECIDING FEASIBILITY

In this section we discuss how to decide feasibility of a given V-RAP instance $\mathcal{I} = \langle G = (U \dot\cup W, E), \mathfrak{S}, c\rangle$. Of the course the feasibility of $\mathcal{I}$ does not depend on the cost vector c. In case $\mathcal{I}$ is not feasible there is a scenario $F \in \mathfrak{S}$ such that the graph $G - F$ has no U-perfect matching. This node set F is a short certificate for the infeasibility of $\mathcal{I}$. Thus, testing feasibility of V-RAP is a problem in coNP. In this section we prove that for V-RAP the feasibility problem is tractable, which is a difference to E-RAP, where the feasibility problem is NP-hard (see Proposition 3.2.1).

Observe that V-RAP is a monotonic problem: Every superset of a feasible solution is feasible itself. Thus, the instance $\mathcal{I}$ is feasible if and only if the entire node set W is a feasible solution. There are two easy cases. If $\mathfrak{S}$ is given explicitly as a list of vulnerable node sets, then we can check if for each scenario $F \in \mathfrak{S}$ the graph $G - F$ has a U-perfect matching using any efficient matching algorithm. Same

holds if $\mathfrak{S} = [W]^{\ell}$, where ℓ is a constant. Hence, in both cases testing feasibility can be performed in time polynomially bounded by the size of the input.

Hence the interesting case appears when the V-RAP instance under consideration is uniform and given as $\langle G, k, c\rangle$, i.e., the scenario set $\mathfrak{S} = [W]^{k}$ is specified implicitly. Then the naive approach described above does not yield an algorithm polynomial in k, because we have to compute $\binom{|W|}{k} = O(|W|^{k})$ matchings. The naive algorithm is just an XP-algorithm with k as parameter.

In the remainder of this section we focus on uniform scenario sets $\mathfrak{S} = [W]^{k}$ given implicitly by providing an integer k as input. We use a result from [Lar+14] for a problem similar to V-RAP that is based on LP techniques. Observe that the ILP formulation (NIAP-ILP) can be extended to V-RAP by introducing a new set of variables for each scenario (cf. the ILP for E-RAP, p. 49). However, it is not obvious how to decide whether the associated polytope has no integer points, i.e., the corresponding instance is feasible.

Laroche et al. [Lar+14] considered a problem related to V-RAP. For a given bipartite graph $G = (U \dot\cup W, E)$ they consider the following interdiction problem that is motivated by the Nurse Rostering Problem arising in health care: Does the removal of k arbitrary nodes from W results in a graph without a U-perfect matching? In the context of health care management this question restates as follows: How many nurses can be absent at most such that all patients can be still treated adequately? Especially, the authors are interested in computing

$$k_{\max} := \max\{|F| \colon F \subseteq W,\ G - F \text{ has a } U\text{-perfect matching}\},$$

i.e., the answer to the second question above [Lar+14, Def. 3]. The parameter $k_{\max}$ can be interpreted as a measure of health care provider's resilience concerning staff unavailability. In particular, this means that unlike V-RAP, their problem is not a design problem.

To tackle their problem the authors exploit the so-called k-extended Hall's condition:

$$\forall\, \emptyset \neq T \subseteq U \colon |T| + k \leqslant |N_G(T)|. \tag{k-Hall}$$

This condition is the natural generalization of the classical Hall's condition (see Theorem 2.3.1) and is for example known in context of k-extendable graphs (see Theorem 2.3.9). It is useful here, as it provides a characterization of k-robust solutions to V-RAP.

Lemma 4.2.1 ([Lar+14, Thm. 2])**.** *Given a bipartite graph $G = (U \dot\cup W, E)$. A set $X \subseteq W$ is k-robust if and only if the k-extended Hall condition is valid for the graph $H := G[U \dot\cup X]$.*

Laroche et al. derived the following identity

$$k_{\max} = \min\{|N_G(T)| - |T| \colon \emptyset \neq T \subseteq U\},$$

from (k-Hall). The minimization problem on the right hand side can be formulated as the following ILP.

$$\begin{aligned} \min \quad & \sum_{w \in W} x_w - \sum_{u \in U} x_u \\ \text{s.t.} \quad & x_u \leqslant x_w && \text{for each } \{u, w\} \in E, \\ & \sum_{u \in U} x_u \geqslant 1, \\ & x \in \{0,1\}^V. \end{aligned} \tag{k_{max}-ILP}$$

The constraint

$$\textstyle\sum_{u \in U} x_u \geqslant 1 \tag{7}$$

ensures that at least one of the nodes in U is chosen by a solution of (k_{max}-ILP). The constraints $x_u \leqslant x_w$ of this ILP can be described by the arc-node incidence matrix of the graph G with edges directed from U to W. Thus, the constraint matrix of (k_{max}-ILP) without Inequality (7) is totally unimodular.

The key idea from [Lar+14] is to replace (7) by $x_u = 1$ for each $u \in U$ and solve the resulting ILP, which is then defined over an integral polytope, via its LP relaxation. The optimal value for the original problem is the minimum of the resulting $|U|$ values. Hence (k_{max}-ILP) can be solved in polynomial time [Lar+14, Cor. 2]. This leads immediately to the following result.

Corollary 4.2.2. *Feasibility testing for* k*-uniform* V-RAP $\langle G, k, c \rangle$ *can be performed efficiently.*

Proof. Given an k-uniform instance $\mathcal{I} := \langle G, k, c \rangle$ for V-RAP, we can determine k_{max} by solving (k_{max}-ILP). The instance $\mathcal{I}$ is feasible if and only if $k_{max} \geqslant k$. ■

4.3 COMPLEXITY OF V-RAP

In this section we show the hardness of V-RAP even if the failure scenarios are restricted to singletons. First, we provide the description of a basic reduction from SET COVER to V-RAP and then show that it is an S-reduction.

Lemma 4.3.1. *There is a basic reduction* (f, g) *from* SET COVER *to* NODE-ROBUST ASSIGNMENT *with* $\mathfrak{S} = W$.

Proof. We prove the four properties in Definition 2.2.3 step by step. Let $\mathcal{I} := \langle [k], \mathcal{S} \rangle$ be an arbitrary feasible instance of SET COVER and $\ell := |\mathcal{S}|$.

(B1): We start the construction of the V-RAP instance $\mathcal{I}' := f(\mathcal{I}) = \langle G, \mathfrak{S}, c\rangle$ with the graph G. The graph is obtained by performing two transformation steps.

(T1) For each $i \in [k]$ we introduce a new node u_i and we set $U_{[k]} := \{u_i : i \in [k]\}$. For each $S_j \in \mathcal{S}$ we introduce a node w_{S_j} and set $W_{\mathcal{S}} := \{w_{S_j} : S_j \in \mathcal{S}\}$. Furthermore, the nodes u_i and W_{S_j} are connected via the edge set $E_{SC} := \big\{\{u_i, w_{S_j}\} : i \in S_j, j \in [\ell]\big\}$.

(T2) For each $i \in [k]$, a copy w_i of node u_i is introduced and we set $W_{[k]} := \{w_i : i \in [k]\}$. Furthermore, the nodes u_i and w_i are connected via the edge set $E_{[k]} := \big\{\{u_i, w_i\} : i \in [k]\big\}$.

In Step T1 we encode the structure of the Set Cover instance $\mathcal{I}$. Step T2 ensures the feasibility of the emerging V-RAP instance. The nodes in $W_{\mathcal{S}}$ indicate which sets S_j, $j \in [\ell]$, are chosen from $\mathcal{S}$ to cover the ground set $[k]$ and are used to define the function g.

Applying the two steps yields a graph $G = (U \dot\cup W, E)$, where $U := U_{[k]}$, $W := W_{\mathcal{S}} \cup W_{[k]}$ and $E := E_{SC} \cup E_{[k]}$. We finish the construction of $\mathcal{I}'$ by declaring the scenario set $\mathfrak{S} := W$ and the weights

$$c \in \mathbb{R}_+^W \text{ with } c_w := \begin{cases} 1 & \text{if } w \in W_{\mathcal{S}}, \\ 0 & \text{if } w \in W_{[k]}. \end{cases} \tag{C}$$

Observe that the presented construction leads to a well-defined function $f : \mathfrak{I}_{SCP} \to \mathfrak{I}_{V\text{-}RAP}$. An example of an instance of V-RAP constructed this way is illustrated in Fig. 14.

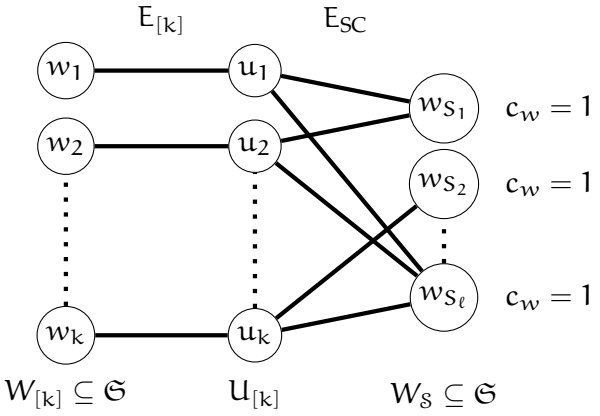

Figure 14: A V-RAP instance $\langle G, W, c\rangle$ constructed by the basic reduction (f, g) from an SCP instance $\langle [k], \{S_1, \ldots, S_\ell\}\rangle$ (see Lemma 4.3.1). The nodes in $W_{[k]}$ and $U_{[k]}$ have zero cost.

(B2): Let $\mathcal{C}$ be a feasible cover for $\mathcal{I}$. We claim

$$X_{\mathcal{C}} := W_{[k]} \cup \{w_{S_j} : S_j \in \mathcal{C},\ j \in [\ell]\}$$

is feasible for $\mathcal{I}'$. Recap that X is feasible to $\mathcal{I}'$ if and only if $H := G[U \dot\cup X] - \{w\}$ contains a U-perfect matching, for each $w \in \mathfrak{S} = W_{[k]} \cup W_{\mathcal{S}}$. To verify the latter condition, consider $w_{S_j} \in W_{\mathcal{S}} \cap \mathfrak{S}$ for some $j \in [\ell]$. Because $W_{[k]} \subseteq H$, we can select the edge set $E_{[k]} = \{\{u_i, w_i\} \colon i \in [k]\}$ as a U-perfect matching in H. Secondly, let $w_i \in \mathfrak{S} \cap W_{[k]}$. Since $\mathcal{C}$ is a feasible cover, the edge $\{u_i, w_{S_j}\}$ is part of $E(H)$. Observe now that

$$\{u_i, w_{S_j}\} \cup E_{[k]} \setminus \big\{\{w_i, u_i\}\big\}$$

is a U-perfect matching in H.

(B3): Let $X' \in \mathtt{sol}(\mathcal{I}')$ be an arbitrary solution to V-RAP. We define $g(\mathcal{I}, X')$ as

$$\mathcal{C}_{X'} := \{S_j \in \mathcal{S} \colon w_{S_j} \in X'\}.$$

We now have to prove that $\mathcal{C}_{X'}$ is feasible for $\mathcal{I}$. Let $i \in [k]$. Consider the scenario in $\mathfrak{S}$ defined by node $w_i \in W_{[k]}$. As X' is feasible to $\mathcal{I}'$, there must exist a U-perfect matching M in $G[U \dot\cup X'] - \{w_i\}$. Since $w_i \notin V(M)$, node u_i is matched to some node in $\{w_{S_1}, \dots, w_{S_\ell}\}$ via edges in E_{SC}. Therefore, for some $S_j \in \mathcal{S}$ with $i \in S_j$ we have $w_{S_j} \in X'$. This implies $S_j \in \mathcal{C}_{X'}$, i.e., the element i is covered by $\mathcal{C}_{X'}$. Hence, $\mathcal{C}_{X'}$ is a feasible cover for $\mathcal{I}$.

(B4): By construction of G, we have that $|V(G)| = 2k + \ell$ and $|E(G)| = k + \sum_{S_j \in \mathcal{S}} |S_j| \leqslant k + k\ell$, i.e., the input size of G is polynomially bounded by the size of $\mathcal{I}$. Because $\mathfrak{S} = W$ and $c \in \{0,1\}^W$, the function f can be computed in time polynomially bounded by $\mathrm{size}(\mathcal{I})$. Since the function g has simply to go through every node in X', g is also polynomial-time computable as well.

■

We are now ready to prove the main result of this chapter, the hardness of approximation for V-RAP with single edge failures.

Theorem 4.3.2. *For every $\varepsilon > 0$, it is* NP*-hard to approximate* Node-Robust Assignment *with $\mathfrak{S} = W$ to within $(1-\varepsilon)\ln n$, where n is the number of nodes in the underlying bipartite graph.*

Proof. We prove the claim by showing that the basic reduction (f, g) as defined in Lemma 4.3.1 satisfies the two properties of an S-reduction (see Definition 2.2.5). Let $\mathcal{I}$ be a feasible instance of Set Cover and $\mathcal{I}' := f(\mathcal{I}) = \langle G, \mathfrak{S}, c\rangle$ the corresponding V-RAP instance.

(S1): Let X' be a feasible solution to $\mathcal{I}'$ and $\mathcal{C}_{X'} := g(\mathcal{I}, X')$. Recap the definition of the cost function in (C). It follows immediately that $c(X') = c(X' \cap W_{\mathcal{S}}) = |\mathcal{C}_{X'}|$.

(S2): Note that by definition of the reduction (f, g), an optimal V-RAP solution $\mathrm{OPT}(\mathcal{I}')$ is mapped to an optimal cover $\mathrm{OPT}(\mathcal{I})$ which implies that both optimal objective values coincide.

Hence, V-RAP inherits all inapproximability properties of SET COVER stated in Theorem 2.4.4. ■

Recap that S-reductions can be also used to deduce parameterized hardness with respect to the standard parameter as well as APX-hardness. Hence using the results for (2,3)-SCP stated in Theorem 2.4.7 and parameterized hardness for general SET COVER stated in Theorem 2.4.8 we immediately obtain the following consequences.

Corollary 4.3.3. V-RAP *is* APX*-hard and* NP*-hard to approximate to within* $\frac{100}{99}$ *even if the maximum degree of the underlying graph is at most three. Furthermore, the standard parameterization of* V-RAP *in general graphs is* W[2]*-hard.*

4.4 AN $(\ln n + 2)$-APPROXIMATION FOR V-RAP

In this section, we present an $(\ln n + 2)$-approximation algorithm for V-RAP with $\mathfrak{S} \subseteq W$, where $n := |U|$ is the number of jobs in the underlying bipartite graph. Because of Theorem 4.3.2 and Theorem 2.4.4 this algorithm is asymptotically tight, i.e., it matches the theoretical bound up to constant additive factors.

To simplify argumentation we assume that every V-RAP instance $\mathcal{I} := \langle G = (U \dot\cup W, E), \mathfrak{S}, c\rangle$, $\mathfrak{S} \subseteq W$, considered in this section is feasible. This can be verified efficiently as described in Section 4.2.

The basic idea of the algorithm is to start with a U-perfect matching M and then extend M to a feasible solution by solving a SET COVER subproblem approximately. The SCP instance is defined as follows. First identify the jobs in G matched to vulnerable machines. We denote this set by U_{risk}. If a scenario is realized, i.e., $w \in \mathfrak{S}$ is deleted from graph, then some job node $u \in U_{\text{risk}}$ matched to w becomes unsaturated. The size of the initial matching M decreases and we know, because the instance is feasible, that there is at least one augmenting path starting at u and ending in the residual pool of machine nodes $W_{\text{res}} = W \setminus V(M)$. Such a path may include other "risky" nodes beside u and hence covers some subset of U_{risk}. An example is illustrated in Fig. 15. The node a is "risky" and is covered by two machines. The machine 1 covers only job a and the machine 4 covers two jobs, a and b. Iterating over all nodes in U_{risk} yields instance of the SET COVER subproblem.

The algorithm is specified as Algorithm 5 and outlined in more details next. The initial U-perfect matching M is computed in Step 4. Then the algorithm identifies all job nodes that are matched by M

Figure 15: Key idea behind Algorithm 5. Input graph with $\mathfrak{S} = W$ and the initial matching M (blue, on the left) and M-alternating paths from the "risky" job a to the remaining machines 1 and 2 after the failure of the machine 2 (on the right).

with a vulnerable machine node, yielding the set U_{risk}. These nodes are the only nodes to be taken care of, since for all other nodes the matching M already provides an assignment. For each residual machine node w, the algorithm next determines a subset R_w of "risky" job nodes that can be robustified by adding w to the solution (Step 9). Thereafter, a weighted Set Cover instance $\mathcal{J}$ with ground set U_{risk} the collection $\{R_w : w \in W_{\text{res}}\}$ is defined and solved by the greedy approximation algorithm. Finally the algorithm returns the machines determined by the matching M and a solution to the SCP subproblem as a solution to the V-RAP instance $\mathcal{I}$.

Algorithm 5: An $(\ln n + 2)$-approximation for V-RAP

Input: Feas. V-RAP-instance $\mathcal{I} = \langle G = (U \dot\cup W, E), \mathfrak{S} \subseteq W, c\rangle$.
Output: A robust solution X for $\mathcal{I}$.
/* Transfer costs from the nodes to the edges */
1 **foreach** $w \in W$ **do**
2 **foreach** $e \in \delta(w)$ **do** $d_e \leftarrow c_w$
3 **end**
4 $M \leftarrow$ min-cost U-perfect matching w.r.t. the costs $d \in \mathbb{R}^E_+$
5 $U_{\text{risk}} \leftarrow \{u \in U : u$ is matched to a vulnerable node in $M\}$
6 $W^M \leftarrow W \cap V(M)$
7 $W_{\text{res}} \leftarrow W \setminus V(M)$
8 **foreach** $w \in W_{\text{res}}$ **do**
  /* Find risky nodes that can be robustified by w */
9 $R_w \leftarrow \{u \in U_{\text{risk}} : w$ is connected to u via an M-alternating path in $G[U \dot\cup (W^M \cup \{w\})]\}$
10 **end**
11 $\mathcal{J} \leftarrow$ weighted SCP instance with the ground set U_{risk}, collection $\mathcal{S} := \{R_w : w \in W_{\text{res}}\}$ and weights $c'(R_w) := c_w$
12 $\text{ALG}(\mathcal{J}) \leftarrow$ output of the greedy SCP algorithm (Alg. 2)
13 $W^{SC} \leftarrow \{w : R_w \in \text{ALG}(\mathcal{J})\}$
14 **return** $X = W^M \dot\cup W^{SC}$

Two properties of the algorithm needed for the analysis are established by the next two lemmas.

Lemma 4.4.1. *Algorithm 5 outputs a feasible solution and runs in polynomial time.*

Proof. We show the efficiency of the algorithm first. The only part potentially in question is the computation of the paths in Step 9, all other operations can be performed in polynomial time. In order to accomplish that, we need the digraph D induced by the matching M as defined in Definition 2.3.7 for perfect matchings. Observe that a path from $w \in W_{\text{res}}$ to some node in U in the digraph D induces an M-alternating path in G. Hence, with D at hand we need to perform a DFS for each $w \in W_{\text{res}}$ and scan the resulting tree for the nodes in U_{risk}. Note that these M-alternating paths just need to be computed once, the nodes from W_{res}, once added to the intermediate solution by the algorithm, can not act as internal nodes of these paths because they are not matched by M.

We now address feasibility. Let X be the solution returned by Algorithm 5. We have to show that X is feasible to $\mathcal{I}$, i.e., for each vulnerable machine node $f \in \mathcal{S}$, there is a U-perfect matching not using f in the graph $G[U \dot\cup X]$. Note that the matching M computed in Step 4 is contained in $G[U \dot\cup X]$, by construction. Let $f \in \mathcal{S}$. If $f \notin V(M)$ then we can use M. In case $f \in \mathcal{S} \cap V(M)$, we denote by u the job node matched to f in M. By definition, u is a "risky" job node, i.e., $u \in U_{\text{risk}}$. From the fact that $\text{ALG}(\mathcal{J})$ forms a cover of U_{risk}, it follows that $u \in R_w$, for some $w \in X \setminus W^M$. By construction of R_w, there exists an odd M-alternating u-w-path P in $G[U \dot\cup (W^M \cup \{w\})]$. As w is not covered by M, the path P ends with an edge incident with w that does not belong to M. Because of this and since P is M-alternating it has an odd number of edges and the first edge of P incident with u is also not contained in M, i.e., $\{u, f\} \notin P$. Thus, $M \triangle (P + \{u, f\})$ is a U-perfect matching in $G[U \dot\cup X] - \{f\}$. ∎

It is not obvious that we can relate the cost of an optimal cover $\text{OPT}(\mathcal{J})$ to the cost of V-RAP's optimum $\text{OPT}(\mathcal{I})$. The machine nodes in $\text{OPT}(\mathcal{I})$ may be very different from the set W^M chosen by the algorithm. For this reason we provide a connection between feasible solutions to V-RAP and Set Cover solutions to $\mathcal{J}$ in the next lemma.

Lemma 4.4.2. *Any feasible solution X for the* V-RAP *instance $\mathcal{I}$ induces a feasible cover $\mathcal{C}_X$ for the instance $\mathcal{J}$ as defined in Step 11 in Algorithm 5, i.e.,* $c'(\text{OPT}(\mathcal{J})) \leqslant c'(\mathcal{C}_X)$.

Proof. Let $X \subseteq W$ be a feasible solution to $\mathcal{I}$. We have to show that $\mathcal{C}_X := \{R_w : w \in X \cap W_{\text{res}}\}$ is a cover for $\mathcal{J}$. Recap that, W_{res} was defined as $W \setminus W^M$, where M is the U-perfect matching computed by the algorithm.

Given an arbitrary $u \in U_{\text{risk}}$ we have to show that $\mathcal{C}_X$ covers u. Let f be the vulnerable node such that $\{u, f\} \in M$. We consider the matching $\hat{M} := M \setminus \{\{u, f\}\}$ of size $|U| - 1$. As X is feasible, the subgraph $H := G[U \dot\cup X] - \{f\}$ contains a U-perfect matching. We denote this matching by N. Because $M \not\subseteq E(H)$, we know $N \neq M$. Let $L := \hat{M} \triangle N$. Because u is only matched by N there is an $\hat{M}$-alternating path P in L that begins at u. Since $|\hat{M}| < |N|$, the path P ends with a node $\bar{w} \in W_{\text{res}}$. Note that all internal nodes of P (if existent) are matched by $\hat{M}$, hence P is a path in the graph H (see Fig. 16). Note that the path P may have length one, i.e., it is just the edge $\{u, \bar{w}\}$ from N. As $\hat{M} \subseteq M$, P is also an M-alternating path, concluding that $u \in R_{\bar{w}}$ and $\mathcal{C}_X$ is a cover.

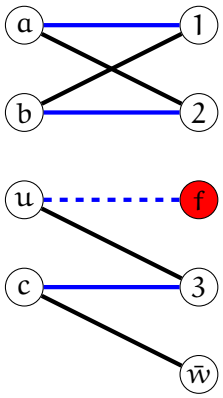

Figure 16: Illustration of the situation in Lemma 4.4.2 after the failure of machine f : the set $X = \{1, 2, 3, \bar{w}\}$, matching $\hat{M}$ (blue) and N (black).

■

We are now equipped to analyze the Algorithm 5 with respect to its approximation quality proving the main algorithmic result of this chapter.

Theorem 4.4.3. *Algorithm 5 is an* $(\ln|U| + 2)$*-approximation for* V-RAP *with* $\mathcal{G} \subseteq W$.

Proof. The question of feasibility and efficiency is already resolved by Lemma 4.4.1. It remains to prove that the computed solution X satisfies the desired quality, i.e., $c(X) \leqslant (\ln|U| + 2) \cdot c(\text{OPT}(\mathcal{I}))$. The cost associated with X are given by

$$c(X) = c(W^M) + c(W^{SC}). \tag{8}$$

As M is chosen in Step 4 as a min-cost U-perfect matching, the associate cost of all covered machine nodes W^M is minimized. Every feasible solution and, hence, every optimal solution contains a U-perfect matching. This implies $|\text{OPT}(\mathcal{I})| \geqslant n$ and we obtain that

$$c(W^M) \leqslant c(\text{OPT}(\mathcal{I})). \tag{9}$$

The classical approximability result for the greedy algorithm (see Theorem 2.4.2) implies that

$$c(W^{SC}) = c'(ALG(\mathcal{J})) \leqslant (\ln|U_{risk}| + 1) \cdot c'(OPT(\mathcal{J})). \tag{10}$$

As $OPT(\mathcal{I})$ is feasible by definition, Lemma 4.4.2 implies that the associated collection $\mathcal{C} := \{R_w : w \in OPT(\mathcal{I}) \setminus W^M\}$ is a cover for $\mathcal{J}$. Thus, we can bound $c'(OPT(\mathcal{J}))$ and obtain

$$c'(OPT(\mathcal{J})) \leqslant c'(\mathcal{C}) = \sum_{R_w \in \mathcal{C}} c_w = c(OPT(\mathcal{I}) \setminus W^M) \leqslant c(OPT(\mathcal{I})). \tag{11}$$

Combining the results from Equations (8)–(11) and exploiting that $U_{\mathfrak{S}} \subseteq U$, we can finally derive that

$$c(X) \leqslant (\ln|U_{risk}| + 2) \cdot c(OPT(\mathcal{I})) \leqslant (\ln|U| + 2) \cdot c(OPT(\mathcal{I})),$$

showing the desired approximation ratio. ■

Note that the approximation ratio we obtain is actually $\ln|U_{risk}| + 2$. But this expression depends on the choice of the matching M in the course of the algorithm. But we also know, that $|U_{risk}| \leqslant \min\{|U|, |\mathfrak{S}|\}$ leading to an approximation guarantee of

$$\ln(\min\{|U|, |\mathfrak{S}|\}) + 2.$$

We hence obtain a better approximation ratio if the number of vulnerable nodes is significantly smaller than $|U|$.

4.5 COMPLEXITY OF CARD-V-RAP

In this section we prove the APX-hardness of MIN-CARD NODE-ROBUST ASSIGNMENT.

The unweighted version of V-RAP is substantially easier than the general one. Using Algorithm 5 we immediately obtain a 2-approximation: select one machine node for each job node covering it. Combining these nodes with the n machine nodes from the initial matching gives an upper bound of $2n$ for a solution determined this way.

Corollary 4.5.1. MIN-CARD NODE-ROBUST ASSIGNMENT *admits a constant factor approximation, i.e.,* Card-V-RAP $\in$ APX.

We prove APX-hardness by exploiting the results for NODE COVER in cubic graphs stated in Theorem 2.4.7. This rules out the existence of a PTAS, unless P = NP. To obtain an L-reduction from NCP we adjust the reduction from Section 4.3.

Lemma 4.5.2. *There is a basic reduction* $(\bar{f}, \bar{g})$ *from* SET COVER *to* MIN-CARD NODE-ROBUST ASSIGNMENT *with* $\mathfrak{S} = W$.

Proof. We prove the four properties in Definition 2.2.3 step by step. Let $\mathcal{I} := \langle [k], \mathcal{S} \rangle$ be an arbitrary feasible instance of Set Cover and $\ell := |\mathcal{S}|$. We adjust the basic reduction (f, g) from Lemma 4.3.1 to derive a reduction $(\bar{f}, \bar{g})$ tailored for the new context.

(B1): We start with the construction of the Card-V-RAP instance $\mathcal{I}' := \bar{f}(\mathcal{I}) = \langle \bar{G}, \bar{\mathfrak{S}} \rangle$ with the graph G. First we use f to obtain the graph $G = (U \dot\cup W, E)$, where $U := U_{[k]}$, $W := W_{\mathcal{S}} \cup W_{[k]}$ and $E := E_{SC} \cup E_{[k]}$.

In order to be able to control the cost of a solution later on, we need to ensure that the nodes in $W_{[k]}$ are contained in any solution to V-RAP. Subsequently, we apply the following additional transformation step: We extend the edges in $E_{[k]}$ from G to a path of length three.

(T3) For each $i \in [k]$, two further nodes $\bar{u}_i$ and $\bar{w}_i$ are introduced. Consequently, we set $\bar{U}_{[k]} := \{\bar{u}_i : i \in [k]\}$ and $\bar{W}_{[k]} := \{\bar{w}_i : i \in [k]\}$. Then, for each $i \in [k]$, the edges $\{w_i, \bar{u}_i\}$ and $\{\bar{u}_i, \bar{w}_i\}$ are introduced and form the set $\bar{E}_{[k]}$ (cf. Step T6 in the reduction from SCP to E-RAP, p. 56).

Applying Step T3 to G yields a graph $\bar{G} = (\bar{U} \dot\cup \bar{W}, \bar{E})$, where $\bar{U} := U_{[k]} \cup \bar{U}_{[k]}$, $\bar{W} := W_{\mathcal{S}} \cup W_{[k]} \cup \bar{W}_{[k]}$ and $\bar{E} := E_{SC} \cup E_{[k]} \cup \bar{E}_{[k]}$. We finish the construction of $\mathcal{I}'$ by defining the scenario set $\bar{\mathfrak{S}} := \bar{W}$. Observe that the presented construction leads to a well-defined function $\bar{f} \colon \mathfrak{I}_{SCP} \to \mathfrak{I}_{\text{Card-V-RAP}}$. An example of a Card-V-RAP instance $\mathcal{I}'$ constructed by $\bar{f}$ is illustrated in Fig. 17.

(B2): Let $\mathcal{C}$ be a feasible cover for $\mathcal{I}$. We claim that

$$X_{\mathcal{C}} := W_{[k]} \cup \bar{W}_{[k]} \cup \{w_{S_j} : S_j \in \mathcal{C}\}$$

is feasible for $\mathcal{I}'$. This means we have to show that the subgraph $H := \bar{G}[\bar{U} \dot\cup X_{\mathcal{C}}] - \{w\}$ contains a U-perfect matching, for each $w \in \bar{\mathfrak{S}} = \bar{W}$. If a node from $W_{\mathcal{S}}$ fails, then we can take the unique U-perfect matching contained in $\bar{E}_{[k]} \cup E_{[k]}$. This matching is denoted by M. In the case that $w_i \in W_{[k]}$ is removed from $\bar{G}$ we can take $(M \setminus \{\{w_i, u_i\}\}) \cup \{\{u_i, w_S\}\}$, where S is some set covering i in $\mathcal{C}$. In the remaining case, $w = \bar{w}_i$ we use the latter matching and replace the edge $\{\bar{w}_i, \bar{u}_i\}$ by $\{\bar{u}_i, w_i\}$.

(B3): The feasibility of

$$\mathcal{C}_{X'} = \{S_j \in \mathcal{S} : W_{S_j} \in X\} = \bar{g}(\mathcal{I}, X')$$

is maintained because the nodes in $W_{[k]}$ are still vulnerable. For this reason we do not need to change the function g, i.e., $\bar{g} := g$.

(B4): By construction of $\mathcal{I}'$, we have $|V(\bar{G})| = 4k + \ell$ and $|E(\bar{G})| = 3k + \sum_{S_j \in \mathcal{S}} |S_j| \leqslant 3k + k\ell$ and $\bar{\mathfrak{S}} = \bar{W}$. This means $\bar{f}, \bar{g}$ are polynomial-time computable functions.

■

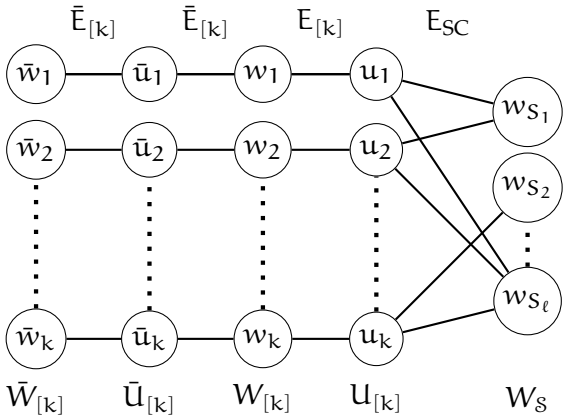

Figure 17: A Card-V-RAP instance $\langle \bar{G}, \bar{W} \rangle$ constructed by the basic reduction $(\bar{f}, \bar{g})$ from an SCP instance $\langle [k], \{S_1, \ldots, S_\ell\} \rangle$ (see Lemma 4.5.2).

Next, we will show that the basic reduction $(\bar{f}, \bar{g})$ above is in fact an L-reduction, thus proving that Card-V-RAP is APX-hard.

Theorem 4.5.3. MIN-CARD NODE-ROBUST ASSIGNMENT *with* $\mathfrak{S} = W$ *is* APX*-complete and* NP*-hard to approximate to within* $\frac{694}{693} \approx 1.0014$*, even in graphs with maximum degree three.*

Proof. We show the claim via a reduction from NODE COVER in cubic graphs. Because 3-NCP is APX-complete (see Theorem 2.4.7) it suffices to provide an L-reduction to prove APX-hardness of Card-V-RAP. Recap that by Observation 2.4.6 3-NCP can be restated equivalently as (2,3)-SCP. For this reason we show that the basic reduction $(\bar{f}, \bar{g})$ defined in Lemma 4.5.2 is an L-reduction when restricted to (2,3)-SCP.

Let $\mathcal{I} = \langle [k], \{S_1, \ldots, S_\ell\} \rangle$ be a feasible instance of (2,3)-SCP and $\mathcal{I}' := \bar{f}(\mathcal{I}) = \langle \bar{G}, \bar{\mathfrak{S}} \rangle$ the corresponding Card-V-RAP instance. Note that the maximum degree in G is bounded by three (see Fig. 17). Let X' be a feasible solution to $\mathcal{I}'$ and $\mathcal{C}_{X'} = \bar{g}(\mathcal{I}, X')$ the associated cover. Recap that the nodes in $\bar{U}_{[k]}$ have degree two in $\bar{G}$, making the inclusion of $\bar{W}_{[k]} \cup W_{[k]}$ into every assignment obligatory. This means

$$|X'| = 2k + |\mathcal{C}_{X'}|.$$

Now we can prove the two properties of an L-reduction.

(L1): The function $\bar{g}$ maps optimal robust assignments to optimal covers. Because we have $|S_j| = 3$, any cover of $[k]$ is of size at least $\frac{k}{3}$, i.e., $k \leqslant 3\mathtt{val}^*(\mathcal{I})$. Hence, we have

$$\mathtt{val}^*(\mathcal{I}') = 2k + \mathtt{val}^*(\mathcal{I}) \leqslant 7\mathtt{val}^*(\mathcal{I}),$$

i.e., the first parameter is $\alpha = 7$.

(L2): The linear relation between both absolute errors is straightforward:

$$|\mathcal{C}_{X'}| - \mathtt{val}^*(\mathcal{I}) = |X'| - 2k - (\mathtt{val}^*(\mathcal{I}') - 2k) = |X'| - \mathtt{val}^*(\mathcal{I}').$$

Hence the second parameter is $\beta = 1$.

Consequently, the reduction $(\bar{f}, \bar{g})$ is an L-reduction with $\alpha = 7$ and $\beta = 1$. Let r' be the approximation ratio for some V-RAP approximation and r the ratio of the approximation for 3-NCP induced by $(\bar{f}, \bar{g})$. We know that $r \leqslant (1 + \alpha\beta(r' - 1))$. Plugging the best known lower bound for 3-NCP of $\frac{100}{99}$ into the latter inequality yields a lower bound on the approximation ratio for Card-V-RAP of $\frac{694}{693}$. Conclusively, the membership in APX follows by Corollary 4.5.1. ■

4.6 CONSTANT-FACTOR APPROXIMATION FOR CARD-V-RAP

We have already seen in Corollary 4.5.1 that Card-V-RAP with $\mathfrak{S} \subseteq W$ admits a 2-approximation. But is an approximation ratio better than 2 possible? In case of uniform instances we can give an affirmative answer.

Theorem 4.6.1. *Algorithm 5 is a 1.75-approximation when restricted to* Card-V-RAP *instances with* $\mathfrak{S} = W$.

Proof. We briefly repeat the basic principle of the algorithm adapted to uniform Card-V-RAP. Here the starting point is any U-perfect matching M since the cost of M is always $|U|$. Further, due to uniformity of $\mathfrak{S}$ all job nodes are matched to a vulnerable machine. Therefore, each job node is "risky" and $U_{\text{risk}} = U$. In the constructed, unweighted Set Cover instance, the algorithm greedily selects the set of machines that covers the most job nodes still in U_{risk}.

To prove the quality of the computed solution X, we proceed as follows. We distinguish two types of iterations performed by the greedy algorithm for SCP. An iteration is called productive if the machine node selected in this iteration saturates at least two "risky" job nodes, i.e., the cardinality of U_{risk} is decreased by at least two. All other iterations are called non-productive. Let p be the total decrease of U_{risk} obtained from all productive iterations combined. We denote by $\text{OPT} \subseteq W$ an optimal solution to the given Card-V-RAP instance. We next show two claims that we use later to derive the approximation ratio of 1.75 for Algorithm 5.

Claim 1: $|X| \leqslant 2|U| - \frac{p}{2}$

The algorithm first includes all machines from the selected matching M into the solution, i.e., $|X| = |W^M| = |U|$. In the uniform case, every job node is "risky". Thus, the ground set of the constructed set cover instance is U, i.e., $|U|$ elements must be taken care of by the greedy algorithm. Since every productive iteration covers at least two

nodes, there can be no more than $\frac{p}{2}$ productive iterations. In each such iteration, one new node is added to X. All remaining iterations are non-productive, and there are exactly $|U| - p$ of them. In total, we have that

$$|X| \leqslant |U| + \frac{p}{2} + (|U| - p) = 2|U| - \frac{p}{2}.$$

Claim 2: $|OPT| \geqslant \max\{|U|, 2(|U| - p)\}$
An optimal solution contains at least one U-perfect matching, i.e., $|OPT| \geqslant |U|$.

Recap from Claim 1 that there are $|U| - p$ non-productive iterations, and that in each non-productive iteration only one "risky" job node is saturated. We denote by U' all job nodes from U being covered in a non-productive iteration of the set cover greedy subroutine. Then, for any pair of distinct nodes $u_1, u_2 \in U'$, $u_1 \neq u_2$, we observe that their neighborhoods in G are disjoint, i.e., $N_G(u_1) \cap N_G(u_2) = \emptyset$. If it would not be the case, the nodes u_1 and u_2 would have been saturated in a productive iteration already, contradicting our assumption $u_1, u_2 \in U'$. Because we have a uniform instance, for any node in U' two neighbors must be included in any feasible solution, i.e., also in OPT. This gives us the inequality $|OPT| \geqslant 2(|U| - p)$. Finally, we derive an upper bound on $\frac{|X|}{|OPT|}$ implying the desired approximation ratio. This is achieved by a case distinction due to Claim 2.

- Case 1: $|OPT| \geqslant |U| \geqslant 2(|U| - p)$, i.e., $2p \geqslant |U|$

  By Claim 1 and the inequality $2p \geqslant |U|$, we obtain that

  $$|X| \leqslant 2|U| - \frac{p}{2} \leqslant 2|U| - \frac{|U|}{4} \leqslant \frac{7}{4}|U| \leqslant \frac{7}{4}|OPT|.$$

- Case 2: $|OPT| \geqslant 2(|U| - p) \geqslant |U|$, i.e., $2p \leqslant |U|$

  By Claim 1 and the inequality $2p \leqslant |U|$, we conclude that

  $$\begin{aligned} |X| &\leqslant 2|U| - \frac{p}{2} \leqslant \frac{7}{2}|U| - \frac{p}{2} - \frac{3}{2}|U| \leqslant \frac{7}{2}|U| - \frac{7}{2}p \\ &\leqslant \frac{7}{2}(|U| - p) \leqslant \frac{7}{4}|OPT|. \end{aligned}$$

■

For non-uniform instances the inequality $|OPT| \geqslant 2(|U| - p)$ in Claim 2 does not hold. The reason is that non-productive iterations from the algorithm depend on the choice of the matching M in Step 4. If the associated job nodes are adjacent with a certain machine node, then in an optimal solution these nodes can be saturated using only one node. This means we only have the bounds $|OPT| \geqslant |U|$ and $|X| \leqslant 2|U| - \frac{p}{2}$ from the analysis of Theorem 4.6.1. Combining these two bounds yields the ratio of 2. But this bound can be obtained using much easier analysis as we have already seen in Corollary 4.5.1.

4.7 CARD-V-RAP WITH TWO SCENARIOS

We complete the complexity landscape for V-RAP by proving that, unlike E-RAP, Card-V-RAP remains tractable with two singleton scenarios. Recap that in Section 4.5, $O(n)$ vulnerable nodes were necessary to derive hardness results.

This result is shown via a reduction to a custom-tailored problem that we can solve efficiently via LP methods.

Theorem 4.7.1. Card-V-RAP *with two vulnerable nodes is solvable in polynomial time.*

Proof. Let $\mathcal{I} = \langle G, \mathfrak{S} \rangle$ be a Card-V-RAP instance with $G = (U \dot\cup W, E)$ and $\mathfrak{S} = \{w', w''\} \subseteq W$, $w' \neq w''$. Given an optimal solution X to $\mathcal{I}$, observe first that either both w' and w'' are contained in X or none of them. In the latter case, an optimal solution is given by a U-perfect matching in $G - \{w', w''\}$. We can use an efficient matching algorithm to verify whether a U-perfect matching in $G - \{w', w''\}$ exists.

In the following, we address the case when $\mathfrak{S}$ is part of any optimal solution. We introduce a dummy job node d and the edges $e' := \{d, w'\}$ and $e'' := \{d, w''\}$. Additionally we double every edge that is not incident with one of the vulnerable nodes w' and w''. This gives us a new graph $G' := (U' \dot\cup W', E')$ where $U' := U \cup \{d\}$, $W' = W$ and $E' := E \cup \{e', e''\} \cup \{\bar e \colon e \in E(U, W \setminus \{w', w''\})$ (see Fig. 18 for an illustration). Note that the new graph G' remains bipartite.

We claim that the solutions to the following ILP correspond to solutions to the V-RAP instance $\mathcal{I}$.

$$
\begin{aligned}
\min \quad & \sum_{e \in E'} x_e & & & \\
\text{s.t.} \quad & x(\delta(u)) &= 2 & \quad \text{for each } u \in U', \\
& x(\delta(w)) &\leqslant 2 & \quad \text{for each } w \in W' \setminus \{w', w''\}, \\
& x_{e'} = x_{e''} &= 1, & \\
& x(\delta(w')) &= 2, & \\
& x(\delta(w'')) &= 2, & \\
& x &\in \{0,1\}^{E'}. &
\end{aligned}
\tag{12}
$$

Every solution to ILP (12) forms a collection of cycles of size at least two and paths that cover every node in U'. Cycles of size two are allowed as we introduced parallel edges in G', i.e., each such cycle represents an original edge of G. The cycle covering node d contains the newly introduced edges e' and e'', and has a size of at least four. This cycle corresponds to a path from w' to w'' in G with an even number of edges. Thus, every solution to ILP (12) defines a union of a w'-w''-path, a (possibly empty) matching, some additional even

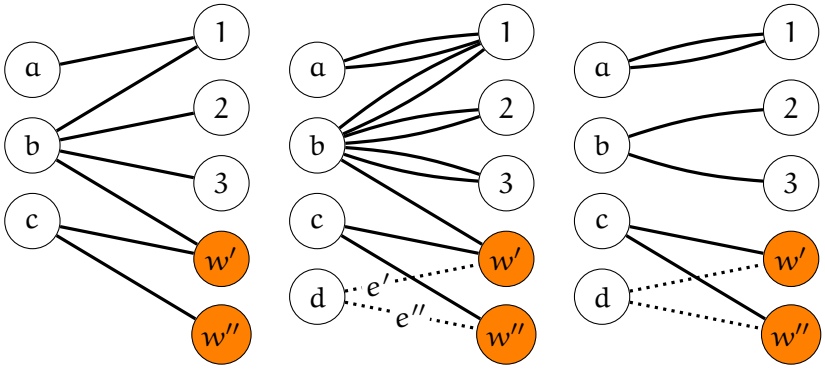

Figure 18: From left to right: Original graph G; graph G′ resulting from the modifications explained in the proof of Theorem 4.7.1; a solution to the corresponding ILP (12) (right).

paths, and potentially some further cycles in the original graph G. The constraint matrix A′ of ILP (12) is essentially the node-edge incidence matrix of G′. As G′ is bipartite, A′ is TU implying that ILP (12) can be solved in polynomial time via LP methods.

First we show that if $\mathcal{I}$ is feasible then ILP (12) has a solution too. Let X be any feasible solution to $\mathcal{I}$. Then, $G[U \dot\cup X]$ contains a matching M' not covering w' as well as a matching M'' not covering w''. By our assumption that $\{w', w''\}$ must be contained in every feasible solution to $\mathcal{I}$, we have that $w'' \in V(M')$ and $w' \in V(M'')$. Then, the symmetric difference $M' \triangle M''$ contains an even path P connecting w' and w''. Moreover, $\hat{M} := M' \setminus E(P)$ is a (possibly) empty matching of the job nodes not covered by the path P. Then, consider $x \in \mathbb{R}^{E'}$ with

$$x_e := \begin{cases} 1, & e \in \{e', e''\} \cup E(P), \\ 1, & e \in \hat{M} \cup \{\bar{e} \colon e \in \hat{M}\}, \\ 0, & \text{otherwise.} \end{cases}$$

By construction the vector x satisfies the constraints of ILP (12).

Now let x be a feasible vertex solution to ILP (12). We set

$$X := \{w \in W \colon \exists e \in \delta_{G'}(w) \text{ with } x_e = 1\},$$

and argue next that X is a robust assignment. To see this, note that each original job node u is adjacent to at least one non-vulnerable machine node from X. Each node $u \in U$ located on a cycle in G′ that is induced by x and does not contain a vulnerable node can be easily matched in $G[U \dot\cup X]$ as u is either incident with an isolated edge in $G[U \dot\cup X]$ or u is part of an even cycle. Even paths in G′ correspond to even paths in G. Since job nodes are internal nodes of these paths, they can be matched using edges from these paths. All remaining nodes in U are internal nodes of the even w'-w''-path induced by x,

i.e., we can find a U-perfect matching in $G[U \dot\cup X]$ not saturating w' and w'' simultaneously. This shows that any feasible solution x of ILP (12) corresponds to a solution X feasible to $\mathcal{I}$.

Solutions to ILP (12) can be used to obtain optimal solutions to $\mathcal{I}$ as we describe next. In the graph $G[U \dot\cup X]$ a job node can be adjacent to two non-vulnerable machines (see Fig. 18). Such solutions are not optimal and can be identified by checking if $G[U \dot\cup X]$ has any odd component not containing w' and w''. In each of these components the number of machine nodes exceeds the number of job nodes by one. By removing an arbitrary machine node from each component we obtain an optimal solution to $\mathcal{I}$. ■

4.8 POLYHEDRAL DESCRIPTION FOR UNIFORM V-RAP

We conclude this chapter by providing an alternative polyhedral description for k-uniform V-RAP. It is defined with the help of a supermodular function and use $|W|$ variables only. Note that it is possible to extend the formulation in (NIAP-ILP), p. 66, to V-RAP by introducing a new set of variables for each scenario (cf. the ILP for E-RAP, p. 49). This approach would require $|\mathfrak{S}||E| + |W|$ variables in total.

We need some definitions first. Let S be a finite set and $f\colon 2^S \to \mathbb{R}$ a function. The function f is supermodular if for any sets $A \subseteq B \subseteq S$ and an element $x \in S \setminus B$ holds

$$f(A \cup \{x\}) - f(A) \leqslant f(B \cup \{x\}) - f(B).$$

If f satisfies $f(A) \leqslant f(B)$ for any sets $A \subseteq B$ then f is called non-decreasing. A contrapolymatroid associated with a supermodular function f is a polyhedron defined as

$$P_f := \left\{ x \in \mathbb{R}^S_+ \colon f(T) \leqslant x(T) \text{ for each } T \subseteq S \right\}. \tag{CPM}$$

Contrapolymatroids have the following properties. P_f is non-empty if and only if $f(\emptyset) \leqslant 0$. For integer-valued f the inequality system of (CPM) is box-TDI and the polyhedron P_f integral. If f is non-decreasing, then we can minimize a linear function over P_f using the greedy algorithm by Edmonds [Edm70]. For a brief introduction we refer the reader to [Sch02, Chap. 44]. Our alternative polyhedral description of the feasible set for k-uniform V-RAP is similar to a contrapolymatroid.

For the definition of the supermodular function we need the notion of critical jobs.

Definition 4.8.1. Let $G = (U \dot\cup W, E)$ be a bipartite graph and $X \subseteq W$ a subset of machines. We call a job $u \in U$ critical with respect to X if

it can only be performed on the machines in X.[1] The set of X-critical jobs is denoted by $\mathrm{crit}_G(X)$ and it holds

$$\mathrm{crit}_G(X) = \{u \in U\colon N_G(u) \subseteq X\}.$$

Lemma 4.8.2. *For a given bipartite graph $G = (U \dot\cup W, E)$ and $k \in \mathbb{Z}_+$, the set function $g\colon 2^W \to \mathbb{R},\ X \mapsto |\mathrm{crit}_G(X)| + k$ is supermodular and non-decreasing.*

Proof. We prove supermodularity first. Note that $g(X \cup \{x\}) - g(X) = |\mathrm{crit}_G(X \cup \{x\})| - |\mathrm{crit}_G(X)|$, so we just need to show the supermodularity of the function $g'(X) := |\mathrm{crit}_G(X)|$.

Consider $A \subseteq B \subseteq W$ and an element $x \in W \setminus B$. The definition of crit_G implies

$$\begin{aligned} g'(A \cup \{x\}) - g'(A) &= \left|\left\{u \in U\colon u \in \mathrm{crit}_G(A \cup \{x\}) \setminus \mathrm{crit}_G(A)\right\}\right| \\ &= \left|\left\{u \in U\colon N_G(u) \subseteq (A \cup \{x\}),\ \{u, x\} \in E\right\}\right| \\ &\leqslant \left|\left\{u \in U\colon N_G(u) \subseteq (B \cup \{x\}),\ \{u, x\} \in E\right\}\right| \\ &= g'(B \cup \{x\}) - g'(B), \end{aligned}$$

thus proving the supermodularity of g'.

Let $A \subseteq B$ be subsets of W and let $u \in \mathrm{crit}_G(A)$. By definition we have $N(u) \subseteq A$ and also $N(u) \subseteq B$. Hence we have $\mathrm{crit}_G(A) \subseteq \mathrm{crit}_G(B)$ implying that $g'(A) \leqslant g'(B)$ and consequently $g(A) \leqslant g(B)$. ∎

We present a characterization of solutions to k-uniform V-RAP using the function g next.

Lemma 4.8.3. *Consider an instance of* V-RAP *on the graph $G = (U \dot\cup W, E)$ with $\mathfrak{S} = [W]^k$. A node set $X \subseteq W$ is k-robust if and only if*

$$\forall\, W' \subseteq W\colon\ \mathrm{crit}_G(W') \neq \emptyset \Rightarrow |\mathrm{crit}_G(W')| + k \leqslant |W' \cap X|.$$

Proof. Let $X \subseteq W$ be a k-robust solution. Due to Lemma 4.2.1 we know that the k-extended Hall's condition is valid for the graph $H := G[U \dot\cup X]$. Consider now a subset $W' \subseteq W$ with $\mathrm{crit}_G(W')$ non-empty. We define $T := \mathrm{crit}_G(W') \subseteq U$. Since G is bipartite we have $N_H(T) = N_G(T) \cap X = W' \cap X$. Now applying k-extended Hall's condition to T and due to the observations above we obtain

$$|\mathrm{crit}_G(W')| + k = |T| + k \leqslant |N_H(T)| = |W' \cap X|.$$

Conversely, let $T \subseteq U$ be non-empty. We define $W' := N_H(T)$, where the subgraph H is again defined as $G[U \dot\cup X]$. By definition it holds

1 The notion of critical jobs here is different from the definition in [Lar+14].

$T \subseteq \operatorname{crit}_G(W')$ and as a result $\operatorname{crit}_G(W')$ is non-empty. Hence we have

$$|T| + k \leqslant |\operatorname{crit}_G(W')| + k \leqslant |W' \cap X|.$$

Since $W' = N_H(T)$ we have $W' \cap X = N_H(T)$ concluding the proof. ■

We can now provide a reformulation of k-uniform V-RAP using the supermodular function $g(W') = |\operatorname{crit}_G(W')| + k$.

Corollary 4.8.4. *The Problem 4.1.1 with the scenario set $\mathfrak{S} = [W]^k$ is equivalent to the following integer linear program*

$$\begin{aligned} \min \quad & c^\top x \\ \text{s.t.} \quad & x(W') \geqslant g(W') \text{ for each } W' \subseteq W \text{ with } \operatorname{crit}_G(W') \neq \emptyset, \\ & x \in \{0,1\}^W. \end{aligned} \tag{13}$$

Let P be the polytope underlying the LP relaxation of (13). The polytope P has no contrapolymatroid structure (see Equation (CPM)), because we omit the inequalities for W' without critical jobs. But this description can be useful for designing branch and bound algorithms in the original space $\mathbb{R}^W$.

For sake of completeness we illustrate the non-integrality of the polytope P in a concrete example next. Recap that in Theorem 4.3.2 we showed that uniform V-RAP is already NP-hard for $k = 1$, which already precludes the integrality of P, provided $P \neq NP$.

Example 4.8.5. Consider the subgraph of $K_{3,6}$ presented in Fig. 19.

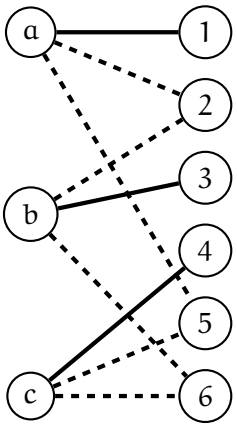

Figure 19: Non-integral vertex solution for the V-RAP formulation from Corollary 4.8.4 with $k = 1$. The edges with values $\frac{1}{2}$ are indicated by dashed lines.

The corresponding polytope P is described by the following non-redundant inequalities

$$
\begin{aligned}
-x_1 - x_2 - x_3 - x_4 - x_5 - x_6 &\leqslant -4,\\
-x_1 - x_2 - x_5 &\leqslant -2,\\
-x_4 - x_5 - x_6 &\leqslant -2,\\
-x_2 - x_3 - x_6 &\leqslant -2,\\
x &\leqslant 1.
\end{aligned}
$$

The point $\bar{x} = \left(1, \frac{1}{2}, 1, 1, \frac{1}{2}, \frac{1}{2}\right)$ is contained in P and hence it describes a fractional solution to V-RAP. In this solution each job in $\{a, b, c\}$ runs on $1 + \frac{1}{2} + \frac{1}{2}$ machines. It is easy to check that the constraint matrix has rank three and also three constraints are tight for $\bar{x}$ and hence it is a vertex solution.

5

ROBUST MATCHING AUGMENTATION

In the previous chapters we studied two robust assignment problems. Both of them are design problems, i.e., given a graph G and a scenario set $\mathfrak{S}$, the task is to find a subgraph containing a perfect matching for each failure scenario in $\mathfrak{S}$. In practice it is not always possible or desired to design an infrastructure from scratch. Building upon a running system is often cheaper and increases acceptance among staff members. In this chapter we address the corresponding augmentation problem: given a fixed perfect matching M in a graph G, we seek to make M fault-tolerant against edge failures by adding new edges to G. Before giving a formal definition in the next section, we present an outline of this chapter first.

OUTLINE

In Section 5.1 we will formally introduce k-Robust s-Recoverable Matching Augmentation (k-s-RRMAP) and explain the two parameters k and s describing the robustness level of a matching. In the subsequent section we treat the basic question of determining how robust a given matching is. This problem will be proved to be NP-hard via a reduction from a new problem, the so-called Fixed Matching Preclusion Number. In Section 5.3 we study the complexity of k-s-RRMAP and prove that even for $k = 1$, the problem is NP-hard. Moreover, the same proof shows, that even in this restricted setting the problem is as hard to approximate as Set Cover. This means that from computational complexity point of view the augmentation problem is as hard as the aforementioned design problems E-RAP and V-RAP. Conclusively in Section 5.4 we study a variant of k-s-RRMAP where $k = 1$ and $s = 2$, which is solvable in polynomial time.

5.1 FORMAL DESCRIPTION AND BASIC PROPERTIES

Prior to defining the augmentation problem sketched above, we first need to specify a reasonable notion of a robust matching. Given a perfect matching M in a graph G we describe its robustness properties by two parameters. The parameter k describes how many edges in M can be removed from G safely, i.e., such that G remains perfectly

matchable. The additional parameter s governs the repairing effort once some edges were removed. This informal definition is summarized next.

Definition 5.1.1. Let G be a bipartite graph. A perfect matching M in G is k-robust s-recoverable if for each subset $F' \subseteq M$, $|F'| \leqslant k$, the graph $G - F'$ has a perfect matching N such that $|M \triangle N| \leqslant 2s$. In other words, the two matchings M and N differ only in s edges. If $s = \infty$, then we call the matching M simply k-robust.

Note that a graph can have perfect matchings with very different levels of robustness (see Fig. 20). Perfect matchings as defined in Definition 5.1.1 were named weakly k-robust s-recoverable in the work by Dourado et al. [Dou+15]. The authors provided a characterization of robust matchings for some values of the parameters k and s.

Theorem 5.1.2 ([Dou+15, Thm. 5])**.** *Let M be a perfect matching in a bipartite graph G. Then, M is k-robust s-recoverable with $k = s \in \{2, 3, 4\}$ if and only if the minimum degree in G is at least* $\max\{2, \frac{|V|}{2} - k + 2\}$.

However, in the main part of [Dou+15] the setting was different. The authors studied the problem to decide whether a given graph G has some k-robust s-recoverable matching and proved that this problem is NP-complete for $k \geqslant 1$ and $s \geqslant 2$ (see p. 28).

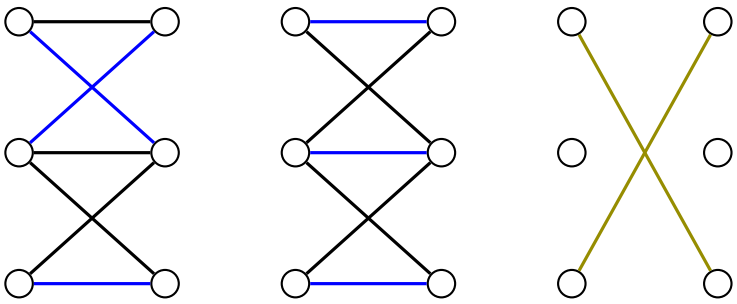

Figure 20: From left to right: a 3-robust matching (blue); a 1-robust matching (blue); two edges (green) needed to increase the robustness level of the second matching.

Our main goal here is to study the problem of increasing the robustness level of a given matching M by adding edges to its ambient graph G. In order to uphold the bipartiteness of the resulting graph, these additional edges must be part of the bipartite complement of $G = (U \dot\cup W, E)$ which is defined as

$$\bar{E}(G) := \big\{\{u, w\} \colon u \in U,\ w \in W\big\} \setminus E(G).$$

We call edges in $\bar{E}(G)$ residual. We are now ready to define the augmentation problem in its full generality.

Problem 5.1.3 (k-s-RRMAP).

k-ROBUST s-RECOVERABLE MATCHING AUGMENTATION	
Instance:	$\langle G, M, R, c, k, s\rangle$, where $G = (U \dot\cup W, E)$ is a bipartite graph, M a perfect matching in G, $R \subseteq \bar{E}(G)$ a set of residual edges, $c \in \mathbb{R}_+^R$, $k \in \mathbb{Z}_+$ and $s \in \mathbb{Z}_+ \cup \{\infty\}$.
Solution:	An augmenting edge set $L \subseteq R$ such that M is a k-robust s-recoverable matching in $G + L$.
Task:	Find L minimizing $c(L)$ or decide that no such L exists.

The augmentation problem k-s-RRMAP with $s = \infty$ can be seen as a special case of E-RAP as defined in Problem 3.1.1, p. 34. Given a k-s-RRMAP instance $\langle G, M, R, c, k, \infty\rangle$, we define the input graph for E-RAP as $G + R$ and the scenario set as $\mathfrak{S} = [M]^k$. Furthermore, all edges except for those in R have zero cost. Then a robust assignment X, feasible for the resulting E-RAP instance, defines an augmenting set $L = X \cap R$. But the results in Chapter 3 do not cover this particular case of E-RAP. Hence, in Section 5.3 we prove that k-s-RRMAP is as hard as SET COVER.

To conclude this section we want to point out two properties of k-s-RRMAP. First, finding an augmenting set $L \subseteq R$ such that a matching M in a graph G is $|M|$-robust in $G + L$ is equivalent to finding a perfect matching in $(G + R) - M$, which is a tractable problem. Hence we can use this augmenting set L to approximate k-s-RRMAP with $s = \infty$ for any $k \geqslant 1$. Because we have $|L| \leqslant \frac{|V(G)|}{2}$ and an optimal solution may have a constant number of edges, this gives us a simple $O(|V(G)|)$-approximation. Second, for a fixed matching M, increasing the robustness level from k to $k+1$ can be the same as increasing k to $|M|$. The matching depicted second in Fig. 20 is 1-robust. In order to make it 2-robust we need both edges shown on the right, but adding these two edges makes M even $|M|$-robust.

5.2 DETERMINING ROBUSTNESS OF A FIXED MATCHING

In this section we study the problem of deciding whether a given perfect matching M in a graph G is k-robust, i.e., for each subset F of M with k elements does $G - F$ has a perfect matching. This question is related to the matching preclusion number $mp(G)$ discussed in the context of feasibility of E-RAP in Section 3.2. Consequently we define the fixed matching preclusion number as

$$\mathrm{fmp}(G, M) := \min\{|F| \colon F \subseteq M, G - F \text{ has no perf. matching}\}. \quad (14)$$

We study the decision version of the latter optimization problem which is defined next.

Problem 5.2.1 (FMPNP).

FIXED MATCHING PRECLUSION NUMBER	
Instance:	$\langle G, M, k\rangle$, where G is a graph, M a perfect matching in G and k an integer.
Question:	Is $\mathrm{fmp}(G, M) \leqslant k$?

We can reformulate FIXED MATCHING PRECLUSION NUMBER as the following interdiction problem using the model introduced in [Zen10] (see p. 29). Let $\langle G, M, k\rangle$ be an instance of FMPNP. The graph G remains unchanged and the edge-weights are uniform, i.e., $w = \mathbb{1}$. The interdiction costs are $c_e = 1$, if $e \in M$ and $c_e = k+1$ if $e \notin M$. The interdiction budget is k. Then $\mathrm{fmp}(G, M) \leqslant k$ if and only if

$$\min\{\nu(G - F) \colon F \subseteq E(G),\ c(F) \leqslant k\} < |M|.$$

The results in [Zen10] do not cover problems of the latter type. Thus we show next, that FMPNP is NP-complete using a reduction from the following problem dealing with sets that violate Hall's condition (see Theorem 2.3.1).

HALL SET (HSP)	
Instance:	$\langle G, k\rangle$, where $G = (U \dot\cup W, E)$ is a bipartite graph and k an integer.
Question:	Is there a set $S \subseteq U$ with $\lvert N_G(S)\rvert < \lvert S\rvert \leqslant k$?

Evidently HALL SET is in NP. Gaspers et al. [Gas+12]. showed that HSP is W[1]-hard when parameterized by k. The parameterized reduction from CLIQUE provided in [Gas+12] is also a many-to-one reduction, hence implying NP-completeness of HSP. Both hardness results carry over to FMPNP as stated in the next theorem.

Theorem 5.2.2. *Problem 5.2.1 is* NP*-complete and its standard parameterization is* W[1]*-hard, even when restricted to bipartite graphs.*

Proof. We provide a many-to-one reduction from HALL SET to FIXED MATCHING PRECLUSION NUMBER in bipartite graphs. Let $\mathcal{I} = \langle G, k\rangle$ be an instance of HALL SET, where G is a bipartite graph on $n + m$ nodes with bipartition $U = \{u_1, \ldots, u_n\}$ and $W = \{w_1, \ldots, w_m\}$. We construct the corresponding instance $\mathcal{I}' = \langle G', M', k'\rangle$ of FMPNP as follows.

(T1) For each edge $e = \{u_i, w_j\} \in E(G)$, $i \in [n]$, $j \in [m]$, we introduce two edges m_i^u and m_j^w. We denote the corresponding edge sets by M^u and M^w, respectively. Then we connect the endpoints of m_i^u and m_j^w by adding two more edges, thus obtaining a 4-cycle. These cycles encode the adjacency relations of G (see Fig. 21 for an example).

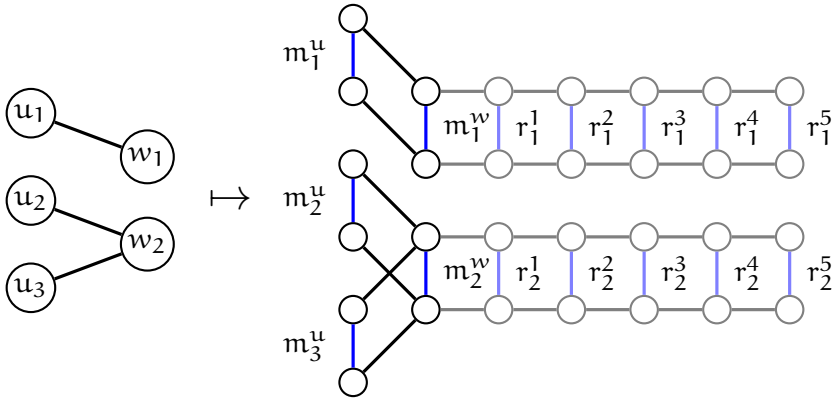

Figure 21: Reduction from Theorem 5.2.2 sketched for the bipartite graph shown on the left. The matching is indicated by blue lines.

(T2) For each $w_j \in W$, $j \in [m]$, we add a ladder graph L_j with rungs $r_j^1, \dots, r_j^{2|U|-1}$ (the lighter subgraph in Fig. 21).

(T3) The matching M' consists of the edge sets M^u, M^w and the rungs of the ladders L_j for $j \in [m]$. Evidently, the matching M' is perfect. The ladders ensure that the robustness of M' is related to the size of the violating Hall set in G.

(T4) We define $k' := k$.

Note that the graph G' is bipartite because it only contains even cycles. Moreover, G' has $O(nm)$ edges, thus the reduction is polynomial in the input size of G. Moreover it is a parameterized reduction. We now show

$$\exists S \subseteq U \text{ satisfying } |N_G(S)| < |S| \text{ with } |S| \leqslant k \Leftrightarrow \operatorname{fmp}(G', M') \leqslant k'.$$

"only if" part:
Let $S \subseteq U$ be a violating Hall set of size at most k in the graph G and $M_S := \{m_i^u : u_i \in S\}$ the corresponding matching in G'. Then $M_S \subseteq M'$ and $|M_S| \leqslant k$. Moreover, $G' - M_S$ does not have a perfect matching by construction of G' (see, e.g., $S = \{u_2, u_3\}$ in Fig. 21). Hence $\operatorname{fmp}(G', M') \leqslant k = k'$.

"if" part:
Let $F' \subseteq M'$, $|F'| \leqslant k'$, be an edge set such that $G' - F'$ has no perfect matching. Because $S \subseteq U$ we can assume $k' = k \leqslant |U|$. Now consider $F^u := F' \cap M^u$. We claim that $G' - F^u$ has no perfect matching. Assume $G' - F^u$ has a perfect matching. Because $G' - F'$ has no perfect matching, there is an edge $m_i^u \in F'$, $i \in [n]$, and an index $j \in [m]$, such that the $|U| + 1$ edges $\{m_i^u, r_j^1, r_j^3, \dots, r_j^{2|U|-1}\}$ are contained in F' (see the top ladder in Fig. 21). Otherwise the deletion of F' can be compensated by shifting the cycles of the ladder graph L_j. But this would imply $|F'| \geqslant |U| + 1$, which contradicts the assumption on the size of F'. Hence $G' - F^u$ has no perfect matching. Define

$S' := \{u \in U \colon m_i^u \in F^u\}$, evidently $|S'| \leqslant k'$. Because $G' - F^u$ has no perfect matching we have $|N_G(S')| < |S'|$. Thus S' is a set violating Hall's condition. ■

Using the result from the preceding theorem we can address our original question of deciding whether a given perfect matching M is k-robust in a graph G. This decision problem is in coNP. If M is not k-robust, then there is an edge set $F \subseteq M$ with $|F| = k$ such that $G - F$ has no perfect matching. This set F is a short certificate for coNP membership.

Corollary 5.2.3. *Let M be a perfect matching in a bipartite graph G. Deciding whether M is k-robust in G is NP-hard.*

Proof. The definition (14) of $\mathrm{fmp}(G, M)$ implies that a perfect matching M is k-robust if and only if $\mathrm{fmp}(G, M) > k$. Thus we can decide FMPNP by applying a robustness oracle to M. ■

5.3 COMPLEXITY OF ROBUST MATCHING AUGMENTATION

In this section we show that k-Robust s-Recoverable Matching Augmentation is as hard as Set Cover even for the special case $s = \infty$ and $c = \mathbb{1}$. This means we study the following problem.

Problem 5.3.1 (1-RMAP)**.**

1-Robust Matching Augmentation	
Instance:	$\langle G, M, R\rangle$, where $G = (U \dot\cup W, E)$ is a bipartite graph, M a perfect matching in G and $R \subseteq \bar{E}(G)$.
Solution:	An augmenting edge set $L \subseteq R$ such that M is a 1-robust perfect matching in $G + L$.
Task:	Find L minimizing $\lvert L\rvert$ or decide that no such L exists.

Observe that a perfect matching M in a bipartite graph G is 1-robust if and only if every M-edge is located on an M-alternating cycle in G. This is the key idea behind the reduction from SCP to 1-RMAP described in Lemma 5.3.2.

First, we define a basic reduction from Set Cover to 1-RMAP which is used in the hardness proof later on.

Lemma 5.3.2. *There is a basic reduction (f, g) from* SCP *to* 1-RMAP.

Proof. We prove the four properties in Definition 2.2.3 step by step. Let $\mathcal{I} := \langle [k], \mathcal{S}\rangle$ be an arbitrary feasible instance of Set Cover and $\ell := |\mathcal{S}|$.

(B1): We start the construction of the 1-RMAP instance $\mathcal{I}' := f(\mathcal{I}) = \langle G, M, R\rangle$ with the graph G. The graph is obtained by performing the following four transformation steps. An example graph is illustrated in Fig. 22.

(T1) Introduce an edge $\{r, x\}$.

(T2) For each $i \in [k]$ introduce an edge $\{u_i, w_i\}$ and connect the nodes u_i to x. The edges $\{u_i, w_i\}$, $i \in [k]$, form the set $E_{[k]}$.

(T3) For each $S_j \in \mathcal{S}$, $j \in [\ell]$, add the nodes $\{S_j, c^1_{S_j}, c^2_{S_j}, c^3_{S_j}\}$ and the 4-cycle $C_{S_j} = (S_j, c^1_{S_j}, c^2_{S_j}, c^3_{S_j}, S_j)$ to G.

(T4) Introduce edges $\{w_i, c^1_{S_j}\}$ if and only if the element $i \in [k]$ is contained in the set $S_j \in \mathcal{S}$.

(T5) The set of residual edges R from the bipartite complement is defined as

$$R := \big\{\{r, S_j\} \subseteq \bar{E}(G) \colon j \in [\ell]\big\}.$$

The described procedure defines a bipartite graph and we fix a bipartition $U \dot\cup W$ by declaring $r \in U$. The designated matching is

$$M := \{r, x\} \cup E_{[k]} \cup \big\{\{S_j, c^1_{S_j}\}, \{c^2_{S_j}, c^3_{S_j}\} \colon j \in [\ell]\big\}.$$

Observe that the presented construction leads to a well-defined function $f \colon \mathfrak{I}_{\text{SCP}} \to \mathfrak{I}_{\text{1-RMAP}}$.

(B2): Recap that the SCP instance $\mathcal{I}$ is feasible and let $\mathcal{C}$ be any feasible cover for $\mathcal{I}$. We claim that $L' := \big\{\{r, S_j\} \colon S_j \in \mathcal{C},\ j \in [\ell]\big\}$ is an augmenting set. To see this, consider an arbitrary element $i \in [k]$ and a set $S_j \in \mathcal{C}$, $j \in [\ell]$, with $i \in S_j$. Then the M-edge $\{u_i, w_i\}$ is located on the M-alternating cycle $(r, x, u_i, w_i, c^1_{S_j}, S_j)$ in $G + L'$.

(B3): Let $L' \in \texttt{sol}(\mathcal{I}')$ be an augmenting set. We define $g(\mathcal{I}, L')$ as

$$\mathcal{C}_{L'} := \big\{S_j \in \mathcal{S} \colon \{r, S_j\} \in L',\ j \in [\ell]\big\}.$$

Let i be an element of the ground set $[k]$. Since L' is feasible, the graph $G + L'$ has an M-alternating cycle C containing $e_i := \{u_i, w_i\}$. By construction of G, there exists no M-alternating cycle in G that contains e_i, hence there is an edge $\{r, S_j\} \in C$, $j \in [\ell]$, with the property $i \in S_j$. This proves that $\mathcal{C}_{L'}$ is a cover for $[k]$.

(B4): The graph G has $2(k+1) + 4\ell$ nodes and $|R| = \ell$, hence f is polynomial time computable. The function g has to iterate over a set of ℓ edges once, thus g can be computed efficiently too.

■

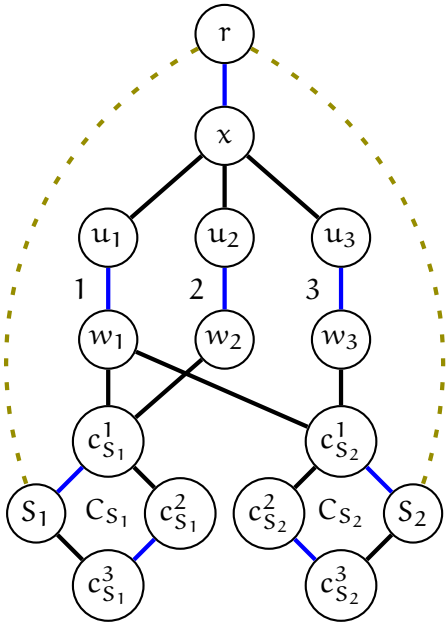

Figure 22: An 1-RMAP instance $\langle G, M, R\rangle$ constructed by the basic reduction (f, g) (see Lemma 5.3.2) from a Set Cover instance $\langle [3], \{S_1, S_2\}\rangle$, where $S_1 = \{1, 2\}$ and $S_2 = \{1, 3\}$. The matching M is indicated by blue edges and the residual edge set R by dashed lines.

We now show the main complexity result of this chapter.

Theorem 5.3.3. *For every $\varepsilon > 0$, it is* NP*-hard to approximate* 1-Robust Matching Augmentation *to within $(1-\varepsilon)\ln n$, where n is the number of nodes in the underlying bipartite graph. Moreover,* 1-RMAP *is* W[2]*-hard when parameterized by the solution size.*

Proof. We show the claim by arguing that the basic reduction (f, g) as defined in Lemma 5.3.2 is in fact an S-reduction (see Definition 2.2.5). Let $\mathcal{I} = \langle [k], \mathcal{S}\rangle$ be a feasible instance of Set Cover with $|\mathcal{S}| = \ell$ and $\mathcal{I}' := f(\mathcal{I}) = \langle G, M, R\rangle$ the corresponding 1-RMAP instance.

(S1): Let $L' \subseteq R$ be a feasible solution to $\mathcal{I}'$. Then, the corresponding cover is $\mathcal{C}_{L'} := g(\mathcal{I}, X') = \{S_j \in \mathcal{S} \colon \{r, S_j\} \in L',\ j \in [\ell]\}$. It follows immediately that $|L'| = |\mathcal{C}_{L'}|$.

(S2): Observe that, by definition of f and g, an optimal 1-RMAP solution $\mathrm{OPT}(\mathcal{I}')$ is mapped to an optimal cover $\mathrm{OPT}(\mathcal{I})$. This implies that both optimal objective values coincide.

Hence, 1-RMAP inherits the inapproximability property of Set Cover stated in Theorem 2.4.4. Furthermore, an S-reduction is also a parameterized reduction with respect to solution size. Hence the standard parameterization of 1-RMAP is $W[2]$-hard by Theorem 2.4.8. ■

Recap that S-reductions can be also used to derive APX-hardness. Hence using the results for (2,3)-SCP stated in Theorem 2.4.7 we arrive at the following result.

Corollary 5.3.4. *In graphs with maximum degree five,* 1-RMAP *is* APX*-hard and* NP*-hard to approximate to within* $\frac{100}{99}$.

Proof. Using an instance of (2,3)-SCP for the reduction yields a graph G with node degrees bounded by 5, except for the node x which has degree $k+1$. To treat this we replace the second and third layer of G by a binary tree of depth $O(\log k)$. It is also possible to choose this tree in a way such that the matching M' can be extended to a matching on the new graph including the tree. Each of the nodes in this tree has degree 2 or 3. ■

It is worth to remark that 1-RMAP remains hard even if we are allowed to choose a perfect matching in the graph that is supposed to be augmented to a 1-robust matching. The graph from the reduction has 2^ℓ perfect matchings, but all of them are Set Cover-hard to augment.

We conclude this section by arguing that the restriction of 1-RMAP (Problem 5.3.1) with the residual edge set $R = \bar{E}(G)$ remains NP-hard to approximate within a sublogarithmic factor. The key observation is that using edges from $\bar{E}(G)$ different from the residual edges $R_s := \{\{r, S\} \in \bar{E}(G) \colon S \in \mathcal{S}\}$ constructed by the basic reduction (f, g) from Lemma 5.3.2, cannot result in strictly smaller augmenting sets. We refer to the edges in R_s as standard-choice edges. This observation is formalized in Lemma 5.3.5. Plugging this result into the proofs of Lemma 5.3.2 and Theorem 5.3.3 yields a reduction $(\bar{f}, \bar{g})$ that is only slightly weaker than an S-reduction because Property S1 from Definition 2.2.5 is satisfied only with inequality as explained next. For any augmenting set $L \subseteq \bar{E}(G)$ for an 1-RMAP instance constructed by $\bar{f}$, we can efficiently obtain a possibly smaller augmenting set L' by following the steps described in the proof of Lemma 5.3.5. Then for the corresponding solution $\mathcal{C}_{L'} := \bar{g}(L')$ to the original Set Cover instance we have $|\mathcal{C}_{L'}| = |L'| \leqslant |L|$ (Property S1 would require $|\mathcal{C}_{L'}| = |L|$). The lemma also implies that the size of the optimal value of a Set Cover instance and the optimal value of the corresponding 1-RMAP instance coincide, i.e., Property S2 remains satisfied for $(\bar{f}, \bar{g})$. In total this shows that an approximation for 1-RMAP would define an approximation for Set Cover with the same approximation guarantee. Hence, due to Theorem 2.4.4 there is no sublogarithmic-factor approximation for 1-RMAP, unless $P = NP$.

Lemma 5.3.5. *Let* $\mathcal{I} = \langle G, M, \bar{E}(G)\rangle$ *be an* 1-RMAP *instance where* G *and* M *are constructed from a feasible* SCP *instance* $\langle [k], \mathcal{S}\rangle$ *by the reduction* (f, g) *defined in Lemma 5.3.2. Then, for any augmenting set* $L \subseteq \bar{E}(G)$ *that*

is feasible for $\mathcal{I}$, there is a feasible augmenting set $L' \subseteq R_s$ with $|L'| \leqslant |L|$ using standard-choice edges R_s only.

Proof. Let $\langle [k], \mathcal{S} \rangle$ be a feasible Set Cover instance and (f, g) the reduction as defined in Lemma 5.3.2. Consider the 1-RMAP instance $\mathcal{I} := \langle G = (U \dot\cup W, E), M, \bar{E}(G) \rangle$, where G and M are constructed using f. Let now $L \subseteq \bar{E}(G)$ be a feasible augmenting set, i.e., every M-edge is part of an M-alternating cycle in the graph $G + L$.

We have to prove that, whenever we use some edges in $\bar{E}(G) \setminus R_s$, we can also use the same number of edges (or even less) in R_s instead. Recap that, because the Set Cover instance $\langle [k], \mathcal{S} \rangle$ we started from is feasible, the edge set R_s is always a feasible augmenting set for $\mathcal{I}$ (see Fig. 22).

It is not obvious if we can replace edges in $\bar{E}(G) \setminus R_s$ by standard-choice edges from R_s in an arbitrary order. For this reason, the proof is carried out in three steps. In the first two steps we replace all non-standard edges contained in L that are adjacent to the nodes r and x. This means concerning r and x we have the situation as illustrated in Fig. 22. More precisely, the proof is organized as follows. Recap that $r \in U$ and $x \in W$. First, we replace all edges in $L \cap \big\{\{r, w\} \in \bar{E}(G) \colon w \in \{w_i \colon i \in [k]\} \cup \{c^2_S \colon S \in \mathcal{S}\} \subseteq W\big\}$ by suitable edges in R_s. Then, in a second step, we replace all edges in $L \cap \big\{\{x, u\} \in \bar{E}(G) \colon u \in \{c^1_S, c^3_S \colon S \in \mathcal{S}\} \subseteq U\big\}$ by edges in R_s. Finally, in a third step we conclude the proof by arguing with the help of an ear decomposition.

Step 1:
We consider two major cases with two subcases each.

a): Let $e = \{r, w_i\} \in L$ for some $i \in [k]$. Fix a set $S \in \mathcal{S}$ with $i \in S$ and define $L' := (L \setminus \{e\}) \cup \{\{r, S\}\}$. We show that every M-edge that is contained in an M-alternating cycle C in $G + L$ using the edge e is part of an M-alternating cycle D in the graph $G + L'$ by providing an explicit description of D.

First, observe that the edge $\{u_i, w_i\} \in M$ is located on the M-cycle $(r, x, u_i, w_i, c^1_S, S, r)$ in $G + L'$. Now, let C be any M-alternating cycle in $G + L$ using $e = \{r, w_i\}$. Because the cycle C is M-alternating, it can be written as

$$C = (x, r, w_i, u_i, p_1, p_2, \dots, p_{2d}, x),$$

where $P := (p_1, p_2, \dots, p_{2d})$, $d \in \mathbb{Z}_+$, is a (possibly empty) M-path in $G + L$ with an even number of nodes. In the first subcase we assume $\{S, c^1_S\} \cap V(P) = \emptyset$. Then the M-cycle

$$D = (x, r, S, c^1_S, w_i, u_i, p_1, p_2, \dots, p_{2d}, x)$$

is contained in $G + L'$. In the second subcase, i.e., $\{S, c^1_S\} \subseteq V(P)$, the cycle C has the form

$$C = (x, r, w_i, u_i, p_1, p_2, \dots, p_{2d'}, S, c^1_S, q_1, q_2, \dots, q_{2d''}, x),$$

where $P := (p_1, p_2, \ldots, p_{2d'})$ and $Q := (q_1, q_2, \ldots, q_{2d''})$, are paths with $d', d'' \in \mathbb{Z}_+$ and $d' + d'' + 1 = d$. Then both M-cycles

$$D' = (u_i, p_1, \ldots, p_{2d'}, S, c_S^1, w_i, u_i) \text{ and}$$
$$D'' = (x, r, S, c_S^1, q_1, q_2, \ldots, q_{2d''}, x)$$

are contained in $G + L'$. Note that the cycle D can not traverse the edge $\{S, c_S^1\}$ in the reversed order, because D is M-alternating. Hence we can replace e by the standard-choice edge $\{r, S\} \in R_s$ without breaking feasibility.

b): Let $e = \{r, c_S^2\} \in L$ for some $S \in \mathcal{S}$. As before, we define $L' := (L \setminus \{e\}) \cup \{\{r, S\}\}$. Consider an M-alternating cycle C in $G + L$ using e. Then C can be written as

$$C = (x, r, c_S^2, c_S^3, p_1, p_2, \ldots, p_{2d}, x),$$

where $P := (p_1, p_2, \ldots, p_{2d})$, $d \in \mathbb{Z}_+$. In the first subcase we assume $S \notin V(P)$. Then the M-cycle

$$D = (x, r, S, c_S^1, c_S^2, c_S^3, p_1, p_2, \ldots, p_{2d}, x)$$

can be used in $G + L'$ instead of C. The second subcase assumes $S \in V(P)$, i.e., the cycle C has the form

$$C = (x, r, c_S^2, c_S^3, p_1, p_2, \ldots, p_{2d'}, S, c_S^1, q_1, q_2, \ldots, q_{2d''}, x),$$

where $d', d'' \in \mathbb{Z}_+$ and $d' + d'' + 1 = d$. Then both M-cycles

$$D' = (c_S^2, c_S^3, p_1, p_2, \ldots, p_{2d'}, S, c_S^2) \text{ and}$$
$$D'' = (x, r, S, c_S^1, q_1, q_2, \ldots, q_{2d''}, x)$$

are contained in $G + L'$. Hence we can replace $e = \{r, c_S^2\}$ by the standard-choice edge $\{r, S\}$ without breaking feasibility.

Step 2:
The arguments are analog to those in Step 1. As before, we have to consider two major cases with two subcases each.

a): Let $e = \{x, c_S^1\} \in L$ for some $S \in \mathcal{S}$. As above, we define $L' := (L \setminus \{e\}) \cup \{\{r, S\}\}$. Consider an M-alternating cycle C in $G + L$ using e. Then C can be written as

$$C = (r, x, c_S^1, S, p_1, p_2, \ldots, p_{2d}, r),$$

where $P := (p_1, p_2, \ldots, p_{2d})$, $d \in \mathbb{Z}_+$, is an M-path in $G + L$.

Recap from the definition of Set Cover (see Problem 2.4.1), that $S \neq \emptyset$. In case there exists $i \in [k]$ with $i \in S$ such that $\{x, u_i, w_i\} \cap V(P) = \emptyset$, use the M-cycle

$$D = (r, x, u_i, w_i, c_S^1, S, p_1, p_2, \ldots, p_{2d}, r)$$

in $G + L'$ instead. Otherwise, we assume that all edges $\{u_i, w_i\}$, $i \in S$, are part of the path P. In case the path P contains only edges $\{u_i, w_i\}$, $i \in S$, there is nothing to do because each of them is part of an M-alternating cycle because of the new edge $\{r, S\}$. Now we assume, without loss of generality, that the edge $e_1 := \{u_1, w_1\} \in E(P)$ and $1 \notin S$. For sake of brevity, we assume the edge e_1 is the only edge from $\big\{\{u_i, w_i\} \colon i \in [k]\big\}$ in P. Then there are elements $a, b \in [k] \cap S$ such that $P' := (u_a, w_a, u_1, w_1, u_b, w_b)$ is a subpath of P. Then we can place the edge $\{u_1, w_1\}$ on the following M-alternating cycle in $G + L'$

$$D = (u_a, x, r, S, c_S^1, w_b, u_b, w_1, u_1, w_a, u_a).$$

Hence we can replace $e = \{x, c_S^1\}$ by a standard-choice edge $\{r, S\}$.

b): Let $e = \{x, c_S^3\} \in L$ and

$$C = (r, x, c_S^3, c_S^2, p_1, p_2, \dots, p_{2d}, r),$$

where $P := (p_1, p_2, \dots, p_{2d})$, $d \in \mathbb{Z}_+$, be an M-cycle in $G + L$. In case $S \in V(P)$, i.e., the cycle C can be written as

$$C = (r, x, c_S^3, c_S^2, p_1, p_2, \dots, p_{2d'}, c_S^1, S, q_1, q_2, \dots, q_{2d''}, r),$$

where $d' + d'' + 1 = d$. Then the path $(p_1, p_2, \dots, p_{2d'})$ is part of the M-cycle

$$D' = (c_S^3, c_S^2, p_1, p_2, \dots, p_{2d'}, c_S^1, S, c_S^3).$$

For the path $(q_1, q_2, \dots, q_{2d''})$ we can repeat the arguments from the preceding subcase. Hence, we only have to show how to treat the path P. In case $S \notin V(P)$, the path P is contained in the M-cycle

$$D = (r, p_1, p_2, \dots, p_{2d}, c_S^2, c_S^3, S, r)$$

in $G + L'$. Hence we can replace $e = \{x, c_S^3\}$ by the standard-choice edge $\{r, S\}$.

The remaining non-standard edges are more complicated to handle and we use more high-level arguments to treat them in the remaining part of the proof. Recap that we have to show that instead of using an arbitrary augmenting set $L \subseteq \bar{E}(G)$ we can use an augmenting set L' using standard-choice edges R_s only and with the property $|L'| \leqslant |L|$.

Let us summarize the situation after the first two steps. We have a feasible augmenting set $L' \subseteq \bar{E}(G)$ and we know that the node r is only adjacent to nodes in $\mathcal{S} \subseteq W$ and x. Furthermore we know that the node x is only adjacent to r and nodes in $\{u_i \colon i \in [k]\} \subseteq U$. This means concerning the nodes r and x we have established the setting as illustrated in Fig. 22. Because L' is feasible we know that there is at least one standard-choice edge $\{r, S\}$, $S \in \mathcal{S}$, contained in the augmenting set L', i.e., the M-edge $\{r, x\}$ is part of some M-cycle in $G + L'$.

Step 3:
We now want to compute an ear decomposition of $G + L'$. First, we remove all dispensable edges from $G + L'$ (see p. 18), which results in a matching-covered graph $\tilde{G}_L$. Observe that, concerning the original edges of G, the removal can only affect the edge sets $\{\{x, u_i\} : i \in [k]\}$ and $\{\{w_i, c_S^1\} : i \in [k],\ S \in \mathcal{S}\}$. Recall, that the removal of dispensable edges disconnects the graph, but every M-edge is still located on some M-alternating cycle in $\tilde{G}_L$ because all edges in those cycles are not dispensable. Note that because of the replacements described in Step 1, the edge $\{r, x\}$ is still located on an M-alternating cycle. Moreover, observe that the cycles C_S, $S \in \mathcal{S}$, are still contained in $\tilde{G}_L$. This means that the only M-edges in $\tilde{G}_L$ after the removal of the non-standard edges, which are possibly not on an M-alternating cycle anymore, are edges in $E_{[k]} = \{\{u_i, w_i\} : i \in [k]\}$. Let $F \subseteq E_{[k]}$ be the set of these edges.

We now have to argue that for each edge in F there is a non-standard edge in $\tilde{G}_L$ and hence we can replace them by edges in R_s one by one. Fix $f \in F$. Let H be a component of $\tilde{G}_L$ with $f \in E(H)$. Because f is on an M-alternating cycle in $\tilde{G}_L$, the component H has at least four nodes. First, in case $\{r, x\} \notin E(H)$, the component H only contains M-edges of the form $\{u_i, w_i\}$, $i \in [k]$, and no edges from the cycles C_S, $S \in \mathcal{S}$. Moreover, H contains a cycle C that spans H and every edge $\{u_i, w_i\} \in E(H) \cap E_{[k]}$ is covered by C. This implies that for every edge $\{u_i, w_i\}$, $i \in [k]$, there is a non-standard edge in H and we are done. Second, if $\{r, x\} \in E(H)$, then let $P_0, \dots, P_q$ be an ear decomposition of H starting with $P_0 = \{r, x\}$ and $M \cap E(H)$ as the initial matching. By construction the edge f is part of some ear P_j, $j \in [q]$. Let $f_1, \dots, f_t$ be the edges in $F \cap E(P_j)$. In case P_j has t non-standard edges we are done. Now assume that P_j contains strictly less than t non-standard edges. Then, P_j must contain edges $\{w_i, c_S^1\}$ and $\{x, u_{i'}\}$ for some $i, i' \in [k]$, $i \neq i'$ and $S \in \mathcal{S}$ (an example with $t = 2$, $i = 2$ and $i' = 3$ is illustrated in Fig. 23). But then the edge $\{u_i, w_i\} \notin F$ because H contains the M-alternating cycle $(u_i, w_i, c_S^1, S, r, x, u_i)$. This contradicts the assumption that we have strictly less than t non-standard edges in P_j.

Thus we know that $\tilde{G}_L$ contains at least $|F|$ non-standard edges and for each $f \in F$ we can replace a non-standard edge by an edge in R_s separately. ■

5.4 AUGMENTING ROBUST RECOVERABLE MATCHINGS

In this section we are returning to the general version of k-ROBUST s-RECOVERABLE MATCHING AUGMENTATION (Problem 5.1.3). Here we study the influence of the recovery parameter s, which controls the size of the M-alternating cycles in the graph.

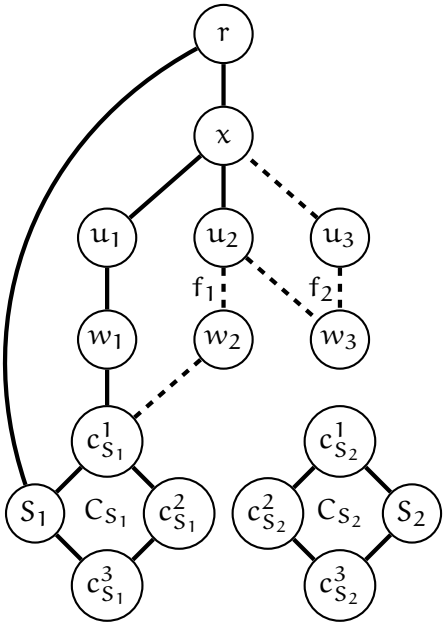

Figure 23: Illustration of the situation in Lemma 5.3.5 for $t = 2$, $i = 2$ and $i' = 3$. The figure shows the ear P_j (dashed) in the matching-covered graph $\tilde{G}_L$ with $f_1 = \{u_2, w_2\}$ and $f_2 = \{u_3, w_3\}$.

First, we argue that even in case $k = 1$ for most values of s the problem is as hard as 1-RMAP. Recap the graph G from the proof of Theorem 5.3.3. First observe that G only has M-alternating cycles of size 4. Second, the addition of standard-choice edges $\{r, S_j\}$, $j \in [\ell]$, puts every edge $\{u_i, w_i\}$ on a cycle of size 6. Hence the reduction (f, g) defined in Lemma 5.3.2 is a reduction from Set Cover to 1-s-RRMAP with $s \geqslant 3$. This immediately gives us the same hardness result as for 1-RMAP.

Theorem 5.4.1. *For every $\varepsilon > 0$ and every $s \geqslant 3$, it is* NP*-hard to approximate* 1-Robust s-Recoverable Matching Augmentation *to within $(1 - \varepsilon) \ln n$, where n is the number of nodes in the underlying bipartite graph. Moreover,* 1-s-RRMAP *is* W[2]*-hard when parameterized by the solution size.*

Surprisingly, 1-2-RMAP is a tractable problem.

Theorem 5.4.2. 1-Robust s-Recoverable Matching Augmentation *with $s = 2$ is solvable in polynomial time.*

Proof. First observe that a matching M in G is 1-robust 2-recoverable if and only if each M-edge is located on an M-alternating cycle of size 4 in G. Therefore, solving 1-2-RMAP is equivalent to adding edges to G to obtain this property.

In order to select the edges accordingly, we use a reduction to the following well-known problem.

MIN-COST EDGE COVER (ECP)	
Instance:	$\langle G, c\rangle$, where $G = (V, E)$ is a graph and $c \in \mathbb{R}_+^E$.
Solution:	A set of edges $E' \subseteq E$, such that $V(E') = V$.
Task:	Find E' minimizing $c(E')$ or decide that no such E' exists.

MIN-COST EDGE COVER can be solved exactly using a reduction to 1-capacitated b-matching [GLS93, p. 259].

Given an 1-2-RMAP instance $\mathcal{I} = \langle G, M, R, c\rangle$, the corresponding ECP instance $\mathcal{I}' = \langle G', c'\rangle$ is defined as follows. Let $n = |M|$ and $m_1, \ldots, m_n$ be the edges in M. The graph G' is constructed by the following two transformation steps.

(T1) For each M-edge m_i, $i \in [n]$, we introduce a node v_i to G'.

(T2) Whenever a pair of M-edges m_i, m_j is located on a 4-cycle in $G + R$ add an edge $\{v_i, v_j\}$ to G'.

In other words, each M-cycle of size 4 in $G + R$ is shrunk to a single edge in G'. To facilitate notation, we extend the cost function $c \in \mathbb{R}_+^R$ to $E(G) \cup R$ by defining $c_e := 0$ for each $e \in E(G)$. Then the cost $c' \in \mathbb{R}_+^{E(G')}$ for the ECP instance is given by

$$c'(\{v_i, v_j\}) = c(C_{ij}), \tag{15}$$

where C_{ij} is the unique 4-cycle in $G + R$ containing the M-edges m_i and m_j. This definition implies, that an edge in G' has zero cost if and only if C_{ij} is a cycle in G.
Note that the reduction can be performed in polynomial time and see Fig. 24 for an example.

Observe that the 1-2-RMAP instance $\mathcal{I}$ is feasible if and only if G' has no isolated nodes, i.e., $\mathcal{I}'$ has a feasible edge cover. We now argue that an optimal solution to $\mathcal{I}'$ is mapped to an optimal solution to $\mathcal{I}$ and both objective values coincide.

First assume $L \subseteq R$ is an augmenting set for the instance $\langle G, M\rangle$. Consider the edge set

$$E_L := \left\{\{v_i, v_j\} \in E(G') \colon L \cap E(C_{ij}) \neq \emptyset\right\} \cup \{e \in E(G') \colon c'_e = 0\}.$$

Because every pair of M-edges is part of a 4-cycle in $G + L$, the edges in E_L form an edge cover for $\mathcal{I}'$.

Conversely, let $E' \subseteq E(G')$ be a feasible solution to the ECP instance $\mathcal{I}' = \langle G', c'\rangle$. The corresponding solution $L' \subseteq R$ for $\langle G, M\rangle$ is given by

$$L' := \{E(C_{ij}) \cap R \colon e' = \{v_i, v_j\} \in E' \text{ and } c'(e') \neq 0\}.$$

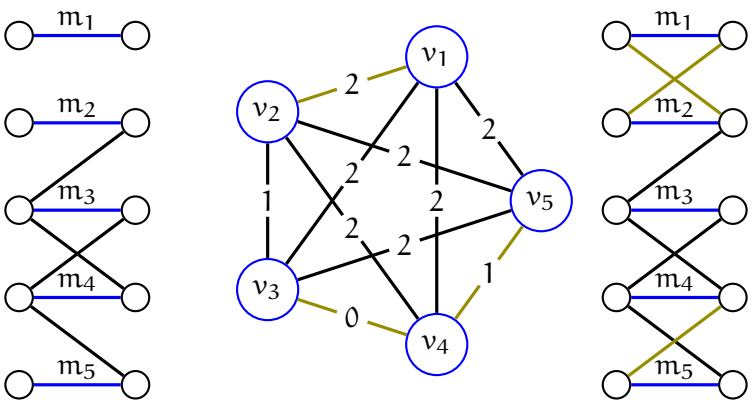

Figure 24: Illustration of the reduction in Theorem 5.4.2. From left to right: an unweighted 1-2-RMAP instance $\langle G, M\rangle$; corresponding ECP instance with a minimum-cost solution E' in G' (green); the input graph G with an optimal augmenting set L for $\langle G, M\rangle$ (green) constructed from the ECP solution E'.

Recap that for each edge $\{v_i, v_j\}$ with non-zero cost, we have that $|R \cap E(C_{ij})| \geqslant 1$. By construction, each M-edge is part of an M-cycle of size 4 in the graph $G + L'$, hence L' is feasible.

In total we have a one-to-one correspondence between solutions to the 1-2-RMAP instance $\mathcal{I}$ and edge covers for $\mathcal{I}'$. Because of (15), the objective values coincide

$$c(L') = \sum_{i,j\in[n]} c(L' \cap E(C_{ij})) = \sum_{\{v_i,v_j\}\in E'} c'(\{v_i, v_j\}) = c'(E').$$

Thus an optimal solution to $\mathcal{I}'$ yields an optimal solution to $\mathcal{I}$. ■

Appendices

A

NP-COMPLETENESS OF BPAFPP

In this section we prove NP-completeness of the special variant of Path Avoiding Forbidden Pairs used in Section 3.3 to prove NP-hardness of Shortest Nice Path. For the sake of clarity we repeat the definition here.

Problem A.1 (BPAFPP).

Path Avoiding Forbidden Pairs in Bipartite Graphs

Instance: $\langle H, s, t, \mathcal{FP}\rangle$, where $H = (U \dot\cup W, E)$ is a balanced bipartite graph, s, t are two distinct nodes in G and a set of node pairs $\mathcal{FP}$ such that the properties

1. $|\mathcal{FP}| = k$ is even,
2. for each pair $(a_i, b_i) \in \mathcal{FP}$: either $a_i, b_i \in U$ or $a_i, b_i \in W$,
3. $\left|\{(a_i, b_i) \in \mathcal{FP} \colon a_i, b_i \in U\}\right| = \left|\{(a_i, b_i) \in \mathcal{FP} \colon a_i, b_i \in W\}\right|$,
4. $s \in U$ and $t \in W$,

hold.

Question: Is there an s-t-path P in G such that for each $(a, b) \in \mathcal{FP}$ at most one of the two nodes is covered by P?

This is a restriction of Path Avoiding Forbidden Pairs to biparite graphs. In directed graphs, the general problem is known to be NP-complete due to Gabow et al. [GMO76, Lemma 2]. Next, we show that our restricted variant of PAFPP remains NP-complete. For this, we adapt the reduction from 3-SAT used in the hardness proof in [GMO76].

Theorem A.2. *The decision problem* BPAFPP *is* NP*-complete.*

Proof. Given a path P in H, it is easy to verify whether P forms a path from s to t and covers at most one node out of $\{a_i, b_i\}$, for each $i \in [k]$. Thus, the problem is in NP.

Now consider an arbitrary 3-SAT instance $\mathcal{I} = \langle B \rangle$, where $B = \bigwedge_{i=1}^{n} C_i$ is a Boolean formula and each clause $C_i = \bigvee_{j=1}^{3} c_{i,j}$ consists of exactly three literals $c_{i,j}$, $j \in [3]$, defined over a finite set of Boolean variables.

Next, we construct an instance $\mathcal{I}'$ of BPAFPP from $\mathcal{I}$. The key idea is that each clause is represented by a "layer" in the graph and the path has to select at least one literal from each clause. The forbidden pairs are defined in a way that this selection yields a valid assignment for B.

The construction of the bipartite graph is described in detail next, an example is shown in Fig. 25. First, we introduce two nodes $v_{i,j}$

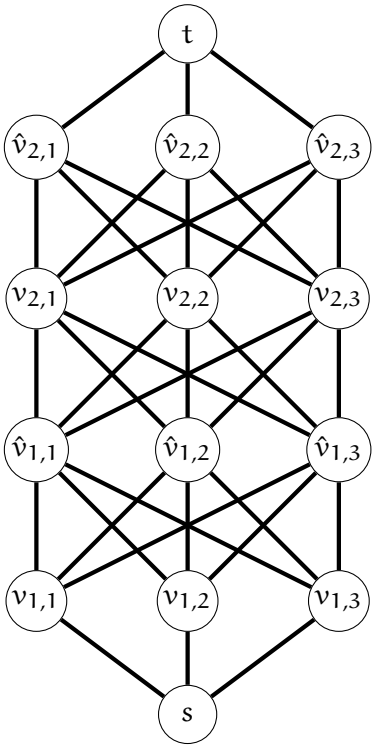

Figure 25: Graph H constructed in reduction from Theorem A.2 for the Boolean formula $(x \vee y \vee z) \wedge (y \vee \bar{x} \vee \bar{z})$. The forbidden pairs are given by $\{(v_{11}, v_{22}), (v_{13}, v_{23}), (\hat{v}_{11}, \hat{v}_{22}), (\hat{v}_{13}, \hat{v}_{23})\}$ corresponding to $(x, \bar{x})$ and $(z, \bar{z})$.

and $\hat{v}_{i,j}$ for each literal $c_{i,j}$ appearing in the Boolean formula B. For each clause $C_i = (c_{i,1} \vee c_{i,2} \vee c_{i,3})$ we construct the complete bipartite graph $(K_{3,3})^i := (V^i \cup \hat{V}^i, E^i)$ with $V^i := \{v_{i,1}, v_{i,2}, v_{i,3}\}$ and $\hat{V}^i := \{\hat{v}_{i,1}, \hat{v}_{i,2}, \hat{v}_{i,3}\}$, i.e., each literal $c_{i,j}$ corresponds to two nodes $v_{i,j}$ and $\hat{v}_{i,j}$. Next, for each $i \in [n-1]$, the bipartite graphs $(K_{3,3})^i$ and $(K_{3,3})^{i+1}$ associated with two consecutive clauses C_i and C_{i+1} (with respect to some arbitrary ordering) are connected to each other by introducing the edge set

$$\bar{E}^i = \big\{\{\hat{v}_{i,j_1}, v_{i+1,j_2}\} \colon j_1, j_2 \in [3]\big\}.$$

This way, we obtain a complete bipartite subgraph on the node set $\{\hat{v}_{i,1}, \hat{v}_{i,2}, \hat{v}_{i,3}\} \dot{\cup} \{v_{i+1,1}, v_{i+1,2}, v_{i+1,3}\}$, $i \in [n-1]$.

Finally, we introduce the nodes s and t. We connect s with each node in V^1, and t with each node in $\hat{V}^n$, i.e., we add the edge sets $E^s := \big\{\{s, v_{1,1}\}, \{s, v_{1,2}\}, \{s, v_{1,3}\}\big\}$ and $E^t := \big\{\{\hat{v}_{n,1}, t\}, \{\hat{v}_{n,2}, t\}, \{\hat{v}_{n,3}, t\}\big\}$. Our transformation yields the bipartite graph $H := (U \dot{\cup} W, E)$ with

$$U := \{s\} \cup \bigcup_{i=1}^{n} \hat{V}^i, \; W := \{t\} \cup \bigcup_{i=1}^{n} V^i \text{ and}$$

$$E := \bigcup_{i=1}^{n} E^i \cup \bigcup_{i=1}^{n-1} \bar{E}^i \cup E^s \cup E^t.$$

To define the set $\mathcal{FP}$ of forbidden pairs we compute the set

$$\text{neg}(B) := \{(c_{i,j}, c_{l,k}) \colon \forall i, l \in [n], \forall j, k \in \{1,2,3\} \text{ with } c_{i,j} = \neg c_{l,k}\},$$

and define $\mathcal{FP} := \big\{(v_{i,j}, v_{l,k}), (\hat{v}_{i,j}, \hat{v}_{l,k}) \colon (c_{i,j}, c_{l,k}) \in \text{neg}(B)\big\}$.

We now claim that $\mathcal{I}' := \langle H, s, t, \mathcal{FP} \rangle$ is an instance of BPAFPP. As $|U| = 1 + 3n = |W|$, the graph H is a balanced, bipartite graph with $s \in U$ and $t \in W$. Moreover, both the number of forbidden pairs $(v_{i,j}, v_{l,k}) \in \mathcal{FP}$, whose nodes are contained in W, and the number of forbidden pairs $(\hat{v}_{i,j}, \hat{v}_{l,k}) \in \mathcal{FP}$, whose nodes are contained in U, equals $|\text{neg}(B)|$. In addition, $|\mathcal{FP}| = 2|\text{neg}(B)|$ is even. Note that the size of $\mathcal{I}'$ is polynomial in size$(\mathcal{I})$.

It remains to show that $\mathcal{I}$ is a Yes-instance of 3-SAT if and only if $\mathcal{I}'$ is a Yes-instance of BPAFPP. We show necessity first. Let P be an s-t path in H that does not cover both nodes of any forbidden pair. Consider now the literals corresponding to nodes in $(W \cap V(P)) \setminus \{t\}$. We denote this set by L. For each clause in B there is at least one literal in L, because the path P has to cross every layer in the graph. Selecting one literal in L per clause and make this literal resolve to `true` gives us a valid assignment for the formula B. Conversely we now take any truth assignment for B. Then we can choose a literal in every clause that causes the clause resolve to `true`. A path from s to t that covers all the nodes corresponding to the chosen literals defines a feasible path for $\mathcal{I}'$. ■

B

NOTES ON E-RAP IN NON-BIPARTITE GRAPHS

Here we sketch two issues that occur if Edge-Robust Assignment is extended to non-bipartite graphs.

B.1 OPTIMAL SOLUTIONS AND k-FACTORS

Recap that by Proposition 3.1.5 (p. 37) we know that if the underlying bipartite graph G has a k-factor, then any k-factor in G is an optimal solution for $(k-1)$-uniform Card-E-RAP instance $\langle G, [E(G)]^{k-1}\rangle$. Unfortunately, this is not longer true for non-bipartite graphs and $k > 2$. The graph G in Fig. 26 is 3-regular but $G - \{4,5\}$ has two odd components and hence G is not $[E(G)]^1$-feasible.

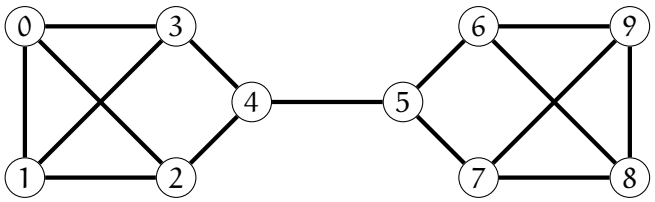

Figure 26: A 3-regular graph which is not 1-uniform feasible.

In case $k = 2$, a k-factor is a collection of cycles and is feasible if and only if every cycle in the collection is even.

For non-bipartite graphs the sufficient condition for feasibility of a k-factor was proved by Plesník: If $X \subseteq E(G)$ is a k-factor in G and each component of $G[X]$ is $(k-1)$-edge-connected, then $G[X]$ has a perfect matching after removal of any k edges [Ple72, Thm. 1].

B.2 GENERAL EAR-DECOMPOSITIONS AND TRIVIAL EARS

The constant-factor approximation algorithm for E-RAP in Section 3.7 computes an approximate solution starting from an ear decomposition of the underlying graph. The key observation is that removing trivial ears from any ear decomposition results in a graph which is again matching-covered. As mentioned in Section 2.3.2, non-bipartite matching-covered graphs also posses ear decompositions. Here we show that the same idea does not work for general graphs by provid-

ing a counterexample.

The relationship between matching-covered graphs and feasible solutions to E-RAP with single edge failures was established in Proposition 3.1.3, p. 36. Note that the corresponding proof does not use the bipartiteness of the graph and thus works for general graphs. In order to present a counterexample, we first need to define a non-bipartite ear decomposition precisely. We repeat the definition of an ear first (see p. 16). Let $G = (V, E)$ be a graph and H a subgraph of G. An ear in G with respect to H is an odd path $P = (v_0, \dots, v_\ell)$ in G such that $v_0 \neq v_\ell$ and $\{v_1, \dots, v_{\ell-1}\} \subseteq V(G) \setminus V(H)$. Ears of length one, i.e. $\ell = 1$, are called trivial.

Definition B.2.1 (adapted from [LP86, p. 175])**.** Let G be a matching-covered graph. An ear decomposition of G is a sequence of graphs $G_0, \dots, G_r$ with the following properties.

- The initial graph G_0 consists of a single edge in G.
- Each graph G_i, $i \in [r]$, results from G_{i-1} by adding one or two odd paths and these paths are ears with respect to G_{i-1}.
- Each graph G_i, $i \in [r]$, is matching-covered.
- Each graph G_i, $i \in [r]$, is nice, i.e., $G - V(G_i)$ is perfectly matchable.
- There holds $G = G_0 + \cdots + G_r$.

If an addition of two ears can not be split up in two single ear additions, then we call this pair of ears a double ear. Note that the ear decomposition defined for bipartite graphs in Definition 2.3.5 satisfies this definition. But unlike for bipartite graphs, we need to require the subgraphs G_i, $i \in [r]$, to be matching-covered and nice explicitly. The major difference is, that for non-bipartite graphs we need to add two ears in one step at some point in the construction to ensure that the resulting graph is matching-covered. An easy example is K_4. We can start with any edge and then add a path of length 3 yielding a 4-cycle. The remaining two edges have to be add at once, because adding just one of them results in a graph which is not matching-covered. In general, for a non-bipartite graph at least one addition of two ears is necessary [LP86, p. 185].

We want to stress, that it is non-trivial to show that adding at most two ears in each step is sufficient. This result is known as The Two Ear Theorem [LP86, Thm. 5.4.6.].

We are now ready to present an example illustrating that both trivial single and trivial double ears may be crucial in an ear decomposition and can not be omitted.

Example B.2.2. We consider the graph G illustrated in Fig. 27. One possible ear decomposition is $G_0, \dots, G_4$ where we start with $G_0 = \{1, 2\}$ and the remaining sequence is defined by

$$G_1 := G_0 + (2, 3, 4, 5, 6, 7, 8) \text{ (black)},$$
$$G_2 := G_1 + (2, 5) \text{ (gray)},$$
$$G_3 := G_2 + (1, 3) + (2, 8) \text{ (blue)},$$
$$G_4 := G_3 + (4, 9, 10, 6) \text{ (green)}.$$

Note that adding only one of the ears $(1, 3)$ or $(2, 8)$ to the graph G_2 does not yield a matching-covered graph. To simplify notation we define $P_s := (2, 5)$, $P_d := (1, 3) + (2, 8)$ and $Q := (4, 9, 10, 6)$, i.e.,

$$G = G_1 + P_s + P_d + Q.$$

Both ears, the single ear P_s and the double ear P_d are trivial and hence do not cover any new nodes. But without them, the graph $G_1 + Q$ is not matching-covered. To see this, consider $(G_1 + Q) - \{9, 10\}$, where the nodes $5, 9$ and 10 can not be matched at the same time. In order to be able to add the path Q to G_1 we need both, the single edge ear P_s and the double ear P_d.

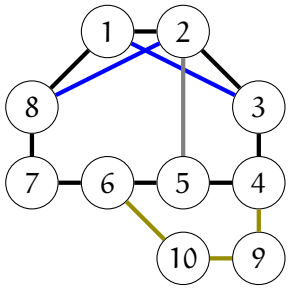

Figure 27: A non-bipartite graph G and an ear decomposition, such that G is not matching-covered if all its trivial ears are removed.

BIBLIOGRAPHY

[Adj17] David Adjiashvili. "Beating Approximation Factor Two for Weighted Tree Augmentation with Bounded Costs." In: *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*. Society for Industrial and Applied Mathematics, 2017, pp. 2384–2399. DOI: 10.1137/1.9781611974782.157 (cit. on p. 31).

[ASZ15] David Adjiashvili, Sebastian Stiller, and Rico Zenklusen. "Bulk-robust combinatorial optimization." *Mathematical Programming* 149.1-2, 2015, pp. 361–390. DOI: 10.1007/s10107-014-0760-6 (cit. on pp. 1, 4, 25, 26).

[ABV05] Hassene Aissi, Cristina Bazgan, and Daniel Vanderpooten. "Complexity of the min–max and min–max regret assignment problems." *Operations Research Letters* 33.6, 2005, pp. 634 –640. DOI: 10.1016/j.orl.2004.12.002 (cit. on p. 26).

[ABV09] Hassene Aissi, Cristina Bazgan, and Daniel Vanderpooten. "Min–max and min–max regret versions of combinatorial optimization problems: A survey." *European Journal of Operational Research* 197.2, 2009, pp. 427 –438. DOI: 10.1016/j.ejor.2008.09.012 (cit. on pp. 24, 26).

[AAL07] Robert E. L. Aldred, Richard P. Anstee, and Stephen C. Locke. "Perfect matchings after vertex deletions." *Discrete Mathematics* 307.23, 2007, pp. 3048–3054. DOI: 10.1016/j.disc.2007.03.017 (cit. on p. 30).

[AK00] Paola Alimonti and Viggo Kann. "Some APX-completeness results for cubic graphs." *Theoretical Computer Science* 237.1, 2000, pp. 123 –134. DOI: 10.1016/S0304-3975(98)00158-3 (cit. on p. 21).

[Aru+16] Ashwin Arulselvan, Ágnes Cseh, Martin Groß, David F. Manlove, and Jannik Matuschke. "Matchings with Lower Quotas: Algorithms and Complexity." *Algorithmica*, 2016. DOI: 10.1007/s00453-016-0252-6 (cit. on p. 27).

[Aus+12] Giorgio Ausiello, Pierluigi Crescenzi, Giorgio Gambosi, Viggo Kann, Alberto Marchetti-Spaccamela, and Marco Protasi. *Complexity and Approximation: Combinatorial Optimization Problems and Their Approximability Properties*. Springer, 2012 (cit. on pp. 7, 11).

[BTN99] Aharon Ben-Tal and Arkadi Nemirovski. "Robust solutions of uncertain linear programs." *Operations Research Letters* 25.1, 1999, pp. 1 –13. DOI: 10.1016/S0167-6377(99)00016-4 (cit. on pp. 22, 23).

[BT+04] Aharon. Ben-Tal, Alexander Goryashko, Elana Guslitzer, and Arkadi Nemirovski. "Adjustable robust solutions of uncertain linear programs." *Mathematical Programming* 99.2, 2004, pp. 351–376. DOI: 10.1007/s10107-003-0454-y (cit. on p. 24).

[BBC11] Dimitris Bertsimas, David B. Brown, and Constantine Caramanis. "Theory and Applications of Robust Optimization." *SIAM review* 53, 2011, pp. 464–501. DOI: 10.1137/080734510 (cit. on p. 22).

[BC10] Dimitris Bertsimas and Constantine Caramanis. "Finite Adaptability in Multistage Linear Optimization." *IEEE Transactions on Automatic Control* 55.12, 2010, pp. 2751–2766. DOI: 10.1109/TAC.2010.2049764 (cit. on p. 24).

[BNS13] Dimitris Bertsimas, Ebrahim Nasrabadi, and Sebastian Stiller. "Robust and Adaptive Network Flows." *Operations Research* 61.5, 2013, pp. 1218–1242. DOI: 10.1287/opre.2013.1200 (cit. on p. 26).

[BS03] Dimitris Bertsimas and Melvyn Sim. "Robust discrete optimization and network flows." *Mathematical Programming* 98.1, 2003, pp. 49–71. DOI: 10.1007/s10107-003-0396-4 (cit. on p. 24).

[BS04] Dimitris Bertsimas and Melvyn Sim. "The Price of Robustness." *Operations Research* 52.1, 2004, pp. 35–53. DOI: 10.1287/opre.1030.0065 (cit. on p. 23).

[Bri+05] Robert C. Brigham, Frank Harary, Elizabeth C. Violin, and Jay Yellen. "Perfect-matching preclusion." *Congressus Numerantium* 174, 2005, pp. 185–192 (cit. on pp. 30, 38).

[BP71] Richard A. Brualdi and Hazel Perfect. "Extension of partial diagonals of matrices I." *Monatshefte für Mathematik* 75.5, 1971, pp. 385–397 (cit. on p. 18).

[BK17a] Christoph Buchheim and Jannis Kurtz. "Min–max–min robust combinatorial optimization." *Mathematical Programming* 163.1, 2017, pp. 1–23. DOI: 10.1007/s10107-016-1053-z (cit. on p. 24).

[BK17b] Christoph Buchheim and Jannis Kurtz. *Robust Combinatorial Optimization under Convex and Discrete Cost Uncertainty*. Tech. rep. 2017. URL: http://www.optimization-online.org/DB_HTML/2017/09/6199.html (cit. on p. 24).

[BDM12] Rainer E. Burkard, Mauro Dell'Amico, and Silvano Martello. *Assignment Problems, Revised Reprint*. Society for Industrial and Applied Mathematics, 2012 (cit. on p. 1).

[CC05] Marcelo H. de Carvalho and Joseph Cheriyan. "An O(VE) Algorithm for Ear Decompositions of Matching-Covered Graphs." *ACM Transactions on Algorithms (TALG)* 1.2, 2005, pp. 324–337. DOI: 10.1145/1103963.1103969 (cit. on p. 19).

[CP10] Shiri Chechik and David Peleg. "Robust Fault Tolerant Uncapacitated Facility Location." In: *27th International Symposium on Theoretical Aspects of Computer Science (STACS)*. Vol. 5. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2010, pp. 191–202. DOI: 10.4230/LIPIcs.STACS.2010.2454 (cit. on p. 26).

[Che+09a] Shiri Chechik, Michael Langberg, David Peleg, and Liam Roditty. "Fault-tolerant spanners for general graphs." In: *Proceedings of the 41st Annual ACM Symposium on Theory of Computing (STOC)*. 2009, pp. 435–444. DOI: 10.1145/1536414.1536475 (cit. on p. 26).

[Che+02] Chandra Chekuri, Anupam Gupta, Amit Kumar, Joseph Naor, and Danny Raz. "Building Edge-Failure Resilient Networks." In: *Proceedings of the 9th Conference on Integer Programming and Combinatorial Optimization (IPCO)*. Springer, 2002, pp. 439–456. DOI: 10.1007/3-540-47867-1_31 (cit. on p. 26).

[Che+09b] Eddie Cheng, Linda Lesniak, Marc J. Lipman, and Laszlo Liptak. "Conditional matching preclusion sets." *Information Sciences* 179.8, 2009, pp. 1092–1101. DOI: 10.1016/j.ins.2008.10.029 (cit. on p. 30).

[CSS01] Joseph Cheriyan, András Sebő, and Zoltán Szigeti. "Improving on the 1.5-Approximation of a Smallest 2-Edge Connected Spanning Subgraph." *SIAM Journal on Discrete Mathematics* 14.2, 2001, pp. 170–180. DOI: 10.1137/S0895480199362071 (cit. on pp. 25, 59).

[CC03] Miroslav Chlebík and Janka Chlebíková. "Inapproximability Results for Bounded Variants of Optimization Problems." In: *Proceedings of 14th International Symposium on Fundamentals of Computation Theory (FCT)*. Springer, 2003, pp. 27–38. DOI: 10.1007/978-3-540-45077-1_4 (cit. on p. 21).

[Chr76] Nicos Christofides. *Worst-case analysis of a new heuristic for the travelling salesman problem*. Tech. rep. 388. Carnegie-Mellon University Pittsburgh, Graduate School of Industrial Administration, 1976 (cit. on p. 1).

[Chv79] Vašek Chvátal. "A Greedy Heuristic for the Set-Covering Problem." *Mathematics of Operations Research* 4.3, 1979, pp. 233–235. DOI: 10.1287/moor.4.3.233 (cit. on p. 20).

[CFS91] Pierluigi Crescenzi, C. Fiorini, and Riccardo Silvestri. "A note on the approximation of the max clique problem." *Information Processing Letters* 40.1, 1991, pp. 1 –5. DOI: 10.1016/S0020-0190(05)80002-X (cit. on p. 11).

[Cyg+15] Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michał Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer, 2015 (cit. on pp. 12, 21).

[Dah16] Mirko Dahlbeck. "Ear decompositions of matching-covered graphs." MA thesis. Technische Universität Dortmund, 2016 (cit. on p. 61).

[Dar+11] Andreas Darmann, Ulrich Pferschy, Joachim Schauer, and Gerhard J. Woeginger. "Paths, trees and matchings under disjunctive constraints." *Discrete Applied Mathematics* 159.16, 2011, pp. 1726–1735. DOI: 10.1016/j.dam.2010.12.016 (cit. on p. 32).

[DW06] Vladimir G. Deineko and Gerhard J. Woeginger. "On the robust assignment problem under a fixed number of cost scenarios." *Operations Research Letters* 34.2, 2006, pp. 175 – 179. DOI: 10.1016/j.orl.2005.04.003 (cit. on p. 26).

[Dha+05] Kedar Dhamdhere, Vineet Goyal, R. Ravi, and Mohit Singh. "How to pay, come what may: approximation algorithms for demand-robust covering problems." In: *46th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*. IEEE Computer Society, 2005, pp. 367–376. DOI: 10.1109/SFCS.2005.42 (cit. on p. 25).

[Die00] Reinhard Diestel. *Graph Theory*. 2nd ed. 173. Springer, 2000 (cit. on p. 5).

[DK11] Michael Dinitz and Robert Krauthgamer. "Fault-tolerant Spanners: Better and Simpler." In: *Proceedings of the 30th Annual ACM SIGACT-SIGOPS Symposium on Principles of Distributed Computing (PODC)*. ACM, 2011, pp. 169–178. DOI: 10.1145/1993806.1993830 (cit. on p. 26).

[DS14] Irit Dinur and David Steurer. "Analytical Approach to Parallel Repetition." In: *Proceedings of the Forty-sixth Annual ACM Symposium on Theory of Computing (STOC)*. ACM, 2014, pp. 624–633. DOI: 10.1145/2591796.2591884 (cit. on pp. 20, 21).

[Dou+15] Mitre Costa Dourado, Dirk Meierling, Lucia D. Penso, Dieter Rautenbach, Fabio Protti, and Aline Ribeiro de Almeida. "Robust recoverable perfect matchings." *Networks* 66.3, 2015, pp. 210–213. DOI: 10.1002/net.21624 (cit. on pp. 28, 38, 90).

[DF99] Rodney G. Downey and Michael R. Fellows. *Parameterized Complexity*. Springer, 1999 (cit. on p. 13).

[Edm65] Jack Edmonds. "Paths, trees, and flowers." *Canadian Journal of mathematics* 17.3, 1965, pp. 449–467 (cit. on p. 14).

[Edm70] Jack Edmonds. "Submodular functions, matroids, and certain polyhedra." In: *Combinatorial Structures and their Applications (Proc. Calgary Internat. Conf., 1969)*. Gordon and Breach, 1970, pp. 69–87 (cit. on p. 84).

[ET76] Kapali P. Eswaran and Robert E. Tarjan. "Augmentation Problems." *SIAM Journal on Computing* 5.4, 1976, pp. 653–665. DOI: 10.1137/0205044 (cit. on p. 31).

[Fav96] Odile Favaron. "On k-factor-critical graphs." *Discuss. Math. Graph Theory* 16.1, 1996, pp. 41–51. DOI: 10.7151/dmgt.1022 (cit. on p. 30).

[Fei98] Uriel Feige. "A Threshold of ln n for Approximating Set Cover." *Journal of the ACM* 45.4, 1998, pp. 634–652. DOI: 10.1145/285055.285059 (cit. on p. 20).

[Fei+07] Uriel Feige, Kamal Jain, Mohammad Mahdian, and Vahab Mirrokni. "Robust Combinatorial Optimization with Exponential Scenarios." In: *Proceedings of the 12th Conference on Integer Programming and Combinatorial Optimization (IPCO)*. Springer, 2007, pp. 439–453. DOI: 10.1007/978-3-540-72792-7_33 (cit. on p. 25).

[Fio+17] Samuel Fiorini, Martin Groß, Jochen Könemann, and Laura Sanità. "A 3/2-Approximation Algorithm for Tree Augmentation via Chvátal-Gomory Cuts." *arXiv preprint*, 2017. URL: https://arxiv.org/abs/1702.05567 (cit. on p. 31).

[FJ81] Greg N. Frederickson and Joseph Ja'Ja'. "Approximation Algorithms for Several Graph Augmentation Problems." *SIAM Journal on Computing* 10.2, 1981, pp. 270–283. DOI: 10.1137/0210019 (cit. on p. 31).

[FKM10] Ryo Fujita, Yusuke Kobayashi, and Kazuhisa Makino. "Robust Matchings and Matroid Intersections." In: *Proceedings of the 18th annual European conference on Algorithms (ESA): Part II*. Springer, 2010, pp. 123–134. DOI: 10.1007/978-3-642-15781-3_11 (cit. on p. 27).

[FM94] Komei Fukuda and Tomomi Matsui. "Finding all the perfect matchings in bipartite graphs." *Applied Mathematics Letters* 7.1, 1994, pp. 15–18. DOI: 10.1016/0893-9659(94)90045-0 (cit. on p. 13).

[GMO76] Harold N. Gabow, Shachindra N. Maheshwari, and Leon J. Osterweil. "On Two Problems in the Generation of Program Test Paths." *IEEE Transactions on Software Engineering* SE-2.3, 1976, pp. 227–231. DOI: 10.1109/TSE.1976.233819 (cit. on p. 107).

[Gab+09] Harold N. Gabow, Michel X. Goemans, Éva Tardos, and David P. Williamson. "Approximating the smallest k-edge connected spanning subgraph by LP-rounding." *Networks* 53.4, 2009, pp. 345–357. DOI: 10.1002/net.20289 (cit. on p. 25).

[GJ79] Michael R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co Ltd, 1979 (cit. on pp. 7, 31, 41).

[Gas+12] Serge Gaspers, Eun Jung Kim, Sebastian Ordyniak, Saket Saurabh, and Stefan Szeider. "Don't Be Strict in Local Search!" In: *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence*. 2012. URL: https://www.aaai.org/ocs/index.php/AAAI/AAAI12/paper/view/4929 (cit. on p. 92).

[Gra+14] Fabrizio Grandoni, R. Ravi, Mohit Singh, and Rico Zenklusen. "New approaches to multi-objective optimization." *Mathematical Programming* 146.1, 2014, pp. 525–554. DOI: 10.1007/s10107-013-0703-7 (cit. on p. 26).

[GLS93] Martin Grötschel, László Lovász, and Alexander Schrijver. *Geometric Algorithms and Combinatorial Optimization*. 2nd ed. Springer, 1993 (cit. on pp. 8, 15, 50, 103).

[HIM03] MohammadTaghi Hajiaghayi, Nicole Immorlica, and Vahab S. Mirrokni. "Power optimization in fault-tolerant topology control algorithms for wireless multi-hop networks." In: *Proceedings of the 9th Annual International Conference on Mobile Computing and Networking (MobiCom)*. ACM, 2003, pp. 300–312. DOI: 10.1145/938985.939016 (cit. on p. 26).

[Han+17] Samuel Haney, Bruce Maggs, Biswaroop Maiti, Debmalya Panigrahi, Rajmohan Rajaraman, and Ravi Sundaram. "Symmetric Interdiction for Matching Problems." In: *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM)*. Vol. 81. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2017, 9:1–9:19. DOI: 10.4230/LIPIcs.APPROX-RANDOM.2017.9 (cit. on p. 29).

[Har70] Darald J. Hartfiel. "A simplified form for nearly reducible and nearly decomposable matrices." *Proceedings of the American Mathematical Society* 24.2, 1970, pp. 388–393 (cit. on p. 17).

[HR02] Refael Hassin and Shlomi Rubinstein. "Robust Matchings." *SIAM Journal on Discrete Mathematics* 15.4, 2002, pp. 530–537. DOI: 10.1137/S0895480198332156 (cit. on p. 27).

[Het64] Gábor Hetyei. "Rectangular configurations which can be covered by 2×1 rectangles." *Pécsi Tan. Foisk. Közl* 8, 1964, pp. 351–367 (cit. on p. 16).

[HK73] John E. Hopcroft and Richard M. Karp. "An $n^{5/2}$ Algorithm for Maximum Matchings in Bipartite Graphs." *SIAM Journal on Computing* 2.4, 1973, pp. 225–231. DOI: 10.1137/0202019 (cit. on p. 14).

[HHS93] Chun-Nan Hung, Lih-Hsing Hsu, and Ting-Yi Sung. "The most vital edges of matching in a bipartite graph." *Networks* 23.4, 1993, pp. 309–313. DOI: 10.1002/net.3230230413 (cit. on p. 28).

[Jai01] Kamal Jain. "A Factor 2 Approximation Algorithm for the Generalized Steiner Network Problem." *Combinatorica* 21.1, 2001, pp. 39–60. DOI: 10.1007/s004930170004 (cit. on p. 26).

[JV00] Kamal Jain and Vijay V. Vazirani. "An Approximation Algorithm for the Fault Tolerant Metric Facility Location Problem." In: *Approximation Algorithms for Combinatorial Optimization (APPROX)*. Springer, 2000, pp. 177–182. DOI: 10.1007/3-540-44436-X_18 (cit. on p. 26).

[Joh74] David S. Johnson. "Approximation algorithms for combinatorial problems." *Journal of Computer and System Sciences* 9.3, 1974, pp. 256 –278. DOI: 10.1016/S0022-0000(74)80044-9 (cit. on p. 20).

[KM08] Rafael R. Kamalian and Vahan V. Mkrtchyan. "On complexity of special maximum matchings constructing." *Discrete Mathematics* 308.10, 2008, pp. 1792–1800. DOI: 10.1016/j.disc.2007.04.029 (cit. on p. 29).

[Kar72] Richard M. Karp. "Reducibility among Combinatorial Problems." In: *Complexity of Computer Computations: Proceedings of a Symposium on the Complexity of Computer Computations*. Springer, 1972, pp. 85–103. DOI: 10.1007/978-1-4684-2001-2_9 (cit. on pp. 19, 21).

[KKZ14] Adam Kasperski, Adam Kurpisz, and Paweł Zieliński. "Recoverable Robust Combinatorial Optimization Problems." In: *Operations Research Proceedings 2012: Selected Papers of the International Annual Conference of the German Operations Research Society (GOR)*. Springer, 2014, pp. 147–153. DOI: 10.1007/978-3-319-00795-3_22 (cit. on pp. 26, 27).

[KZ09] Adam Kasperski and Paweł Zieliński. "On the approximability of minmax (regret) network optimization problems." *Information Processing Letters* 109.5, 2009, pp. 262–266 (cit. on p. 26).

[KZ16] Adam Kasperski and Paweł Zieliński. "Robust Discrete Optimization Under Discrete and Interval Uncertainty: A Survey." In: *Robustness Analysis in Decision Aiding, Optimization, and Analytics*. Springer, 2016, pp. 113–143. DOI: 10.1007/978-3-319-33121-8_6 (cit. on p. 24).

[KKMU08] Irit Katriel, Claire Kenyon-Mathieu, and Eli Upfal. "Commitment under uncertainty: Two-stage stochastic matching problems." *Theoretical Computer Science* 408.2, 2008, pp. 213–223. DOI: 10.1016/j.tcs.2008.08.010 (cit. on p. 28).

[Kha80] Leonid G. Khachiyan. "Polynomial algorithms in linear programming." *USSR Computational Mathematics and Mathematical Physics* 20.1, 1980, pp. 53 –72. DOI: https://doi.org/10.1016/0041-5553(80)90061-0 (cit. on p. 15).

[KN16] Guy Kortsarz and Zeev Nutov. "A Simplified 1.5-Approximation Algorithm for Augmenting Edge-Connectivity of a Graph from 1 to 2." *ACM Transactions on Algorithms (TALG)* 12.2, 2016, 23:1–23:20. DOI: 10.1145/2786981 (cit. on p. 31).

[KY97] Panos Kouvelis and Gang Yu. *Robust discrete optimization and its applications*. Vol. 14. Kluwer Academic Publishers, 1997. DOI: 10.1007/978-1-4757-2620-6 (cit. on pp. 23, 26).

[Kri75] Mukkai S. Krishnamoorthy. "An NP-hard Problem in Bipartite Graphs." *SIGACT News* 7.1, 1975, pp. 26–26. DOI: 10.1145/990518.990521 (cit. on p. 61).

[Kuh55] Harold W. Kuhn. "The Hungarian method for the assignment problem." *Naval Research Logistics Quarterly* 2.1-2, 1955, pp. 83–97. DOI: 10.1002/nav.3800020109 (cit. on p. 14).

[Kut+08] Martin Kutz, Khaled Elbassioni, Irit Katriel, and Meena Mahajan. "Simultaneous matchings: Hardness and approximation." *Journal of Computer and System Sciences* 74.5, 2008, pp. 884–897. DOI: 10.1016/j.jcss.2008.02.001 (cit. on p. 27).

[Lac+12] Mathieu Lacroix, Ali R. Mahjoub, Sébastien Martin, and Christophe Picouleau. "On the NP-completeness of the perfect matching free subgraph problem." *Theoretical Computer Science* 423, 2012, pp. 25 –29. DOI: 10.1016/j.tcs.2011.12.065 (cit. on p. 38).

[LL98] J. Lakhal and L. Litzler. "A polynomial algorithm for the extendability problem in bipartite graphs." *Information Processing Letters* 65.1, 1998, pp. 11–16. DOI: 10.1016/S0020-0190(97)00177-4 (cit. on p. 19).

[Lar+14] Pierre Laroche, Franc Marchetti, Sébastien Martin, and Zsuzsanna Róka. "Bipartite Complete Matching Vertex Interdiction Problem: Application to Robust Nurse Assignment." In: *International Conference on Control, Decision and Information Technologies (CoDIT)*. 2014, pp. 182–187. DOI: 10.1109/CoDIT.2014.6996890 (cit. on pp. 27, 68–70, 85).

[Lie+09] Christian Liebchen, Marco Lübbecke, Rolf Möhring, and Sebastian Stiller. "The Concept of Recoverable Robustness, Linear Programming Recovery, and Railway Applications." In: *Robust and Online Large-Scale Optimization: Models and Techniques for Transportation Systems*. Springer, 2009, pp. 1–27. DOI: 10.1007/978-3-642-05465-5_1 (cit. on p. 24).

[LY93] Jiping Liu and Qinglin Yu. "Matching Extensions and Products of Graphs." In: *Annals of Discrete Mathematics*. Vol. 55. Elsevier, 1993, pp. 191–200. DOI: 10.1016/S0167-5060(08)70389-3 (cit. on p. 30).

[LY04] Dingjun Lou and Qinglin Yu. "Sufficient conditions for n-matchable graphs." *Australasian Journal Of Combinatorics* 29, 2004, pp. 127–134. URL: http://ajc.maths.uq.edu.au/pdf/29/ajc_v29_p127.pdf (cit. on p. 30).

[Lov75] László Lovász. "On the ratio of optimal integral and fractional covers." *Discrete Mathematics* 13.4, 1975, pp. 383 –390. DOI: 10.1016/0012-365X(75)90058-8 (cit. on p. 20).

[LP77] László Lovász and Michael D. Plummer. "On minimal elementary bipartite graphs." *Journal of Combinatorial Theory, Series B* 23.1, 1977, pp. 127 –138. DOI: 10.1016/0095-8956(77)90062-4 (cit. on p. 62).

[LP86] László Lovász and Michael D. Plummer. *Matching theory*. North-Holland Publishing Co., 1986 (cit. on pp. 13, 15–19, 37, 112).

[LY94] Carsten Lund and Mihalis Yannakakis. "On the Hardness of Approximating Minimization Problems." *Journal of the ACM* 41.5, 1994, pp. 960–981. DOI: 10.1145/185675.306789 (cit. on p. 20).

[MMO17] Jannik Matuschke, S. Thomas McCormick, and Gianpaolo Oriolo. "Rerouting Flows When Links Fail." In: *44th International Colloquium on Automata, Languages, and Programming (ICALP)*. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2017, 89:1–89:13. DOI: 10.4230/LIPIcs.ICALP.2017.89 (cit. on p. 26).

[MSS14] Jannik Matuschke, Martin Skutella, and José A. Soto. "Robust randomized matchings." In: *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*. Society for Industrial and Applied Mathematics, 2014, pp. 1904–1915. DOI: 10.1137/1.9781611973730.127 (cit. on p. 27).

[OZP13] Temel Öncan, Ruonan Zhang, and Abraham P. Punnen. "The minimum cost perfect matching problem with conflict pair constraints." *Computers & Operations Research* 40.4, 2013, pp. 920–930. DOI: 10.1016/j.cor.2012.10.022 (cit. on p. 32).

[OM87] Pekka Orponen and Heikki Mannila. *On approximation preserving reductions: Complete problems and robust measures*. Technical Report C-1987-28. Department of Computer Science, University of Helsinki, 1987 (cit. on p. 11).

[PY91] Christos H. Papadimitriou and Mihalis Yannakakis. "Optimization, approximation, and complexity classes." *Journal of Computer and System Sciences* 43.3, 1991, pp. 425 –440. DOI: 10.1016/0022-0000(91)90023-X (cit. on p. 11).

[PI11] Jung-Heum Park and Insung Ihm. "Strong matching preclusion." *Theoretical Computer Science* 412.45, 2011, pp. 6409–6419. DOI: 10.1016/j.tcs.2011.08.008 (cit. on p. 30).

[Ple72] Ján Plesník. "Connectivity of regular graphs and the existence of 1-factors." *Matematický časopis* 22.4, 1972, pp. 310–318 (cit. on pp. 29, 37, 111).

[Plu86] Michael D. Plummer. "Matching extension in bipartite graphs." *Congressus Numerantium* 54, 1986, pp. 245–258 (cit. on p. 18).

[Plu94] Michael D. Plummer. "Extending matchings in graphs: A survey." *Discrete Mathematics* 127.1–3, 1994, pp. 277–292. DOI: 10.1016/0012-365X(92)00485-A (cit. on p. 19).

[PA96] Michael I. Porteous and Robert E. L. Aldred. "Matching extensions with prescribed and forbidden edges." *Australasian Journal Of Combinatorics* 13, 1996, pp. 163–174 (cit. on p. 30).

[Rég94] Jean-Charles Régin. "A Filtering Algorithm for Constraints of Difference in CSPs." In: *Proceedings of the Twelfth National Conference on Artificial Intelligence (Vol. 1)*. American Association for Artificial Intelligence, 1994, pp. 362–367 (cit. on p. 18).

[Sch98] Alexander Schrijver. *Theory of Linear and Integer Programming*. John Wiley & Sons, 1998 (cit. on p. 14).

[Sch02] Alexander Schrijver. *Combinatorial Optimization: Polyhedra and Efficiency*. Springer, 2002 (cit. on p. 84).

[SV14] András Sebő and Jens Vygen. "Shorter tours by nicer ears: 7/5-Approximation for the graph-TSP, 3/2 for the path version, and 4/3 for two-edge-connected subgraphs." *Combinatorica* 34.5, 2014, pp. 597–629. DOI: 10.1007/s00493-014-2960-3 (cit. on p. 59).

[ŞAO09] Onur Şeref, Ravindra K. Ahuja, and James B. Orlin. "Incremental Network Optimization: Theory and Algorithms." *Operations Research* 57.3, 2009, pp. 586–594. DOI: 10.1287/opre.1080.0607 (cit. on p. 31).

[SS93] Edwin Hsing-Mean Sha and Kenneth Steiglitz. "Maintaining bipartite matchings in the presence of failures." *Networks* 23.5, 1993, pp. 459–471. DOI: 10.1002/net.3230230503 (cit. on p. 32).

[Sla96] Petr Slavík. "A Tight Analysis of the Greedy Algorithm for Set Cover." In: *Proceedings of the Twenty-eighth Annual ACM Symposium on Theory of Computing (STOC)*. ACM, 1996, pp. 435–441. DOI: 10.1145/237814.237991 (cit. on p. 20).

[Soy73] Allen L. Soyster. "Technical Note—Convex Programming with Set-Inclusive Constraints and Applications to Inexact Linear Programming." *Operations Research* 21.5, 1973, pp. 1154–1157. DOI: 10.1287/opre.21.5.1154 (cit. on p. 22).

[SS03] Chaitanya Swamy and David B. Shmoys. "Fault-tolerant Facility Location." In: *Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*. Society for Industrial and Applied Mathematics, 2003, pp. 735–736. URL: http://dl.acm.org/citation.cfm?id=644108.644228 (cit. on p. 26).

[Tar86] Éva Tardos. "A Strongly Polynomial Algorithm to Solve Combinatorial Linear Programs." *Operations Research* 34.2, 1986, pp. 250–256. DOI: 10.1287/opre.34.2.250 (cit. on p. 15).

[Tut54] William T. Tutte. "A short proof of the factor theorem for finite graphs." *Canad. J. Math* 6.1954, 1954, pp. 347–352 (cit. on p. 14).

[VV16] Carlos E. Valencia and Marcos C. Vargas. "Optimum matchings in weighted bipartite graphs." *Boletín de la Sociedad Matemática Mexicana* 22.1, 2016, pp. 1–12. DOI: 10.1007/s40590-015-0065-7 (cit. on p. 18).

[Val79] L.G. Valiant. "The complexity of computing the permanent." *Theoretical Computer Science* 8.2, 1979, pp. 189 –201. DOI: 10.1016/0304-3975(79)90044-6 (cit. on p. 13).

[WYZ09] Xiumei Wang, Jinjiang Yuan, and Sujing Zhou. "Edge-deletable IM-extendable graphs with minimum number of edges." *Discrete Mathematics* 309.16, 2009, pp. 5242 –5247. DOI: 10.1016/j.disc.2009.03.048 (cit. on p. 30).

[Whi32] Hassler Whitney. "Non-separable and planar graphs." *Transactions of the American Mathematical Society* 34.2, 1932, pp. 339–362 (cit. on p. 16).

[WS11] David P. Williamson and David B. Shmoys. *The Design of Approximation Algorithms*. Cambridge University Press, 2011, pp. 1–504 (cit. on p. 55).

[Woe07] Gerhard J. Woeginger. "Match, match, match and match again." *OPTIMA* 73, 2007, pp. 6–8 (cit. on p. 1).

[Zen10] Rico Zenklusen. "Matching interdiction." *Discrete Applied Mathematics* 158.15, 2010, pp. 1676 –1690. DOI: 10.1016/j.dam.2010.06.006 (cit. on pp. 29, 92).

[Zen+09] Rico Zenklusen, Bernard Ries, Christophe Picouleau, Dominique De Werra, Marie-Christine Costa, and Cédric Bentz. "Blockers and transversals." *Discrete Mathematics* 309.13, 2009, pp. 4306–4314. DOI: 10.1016/j.disc.2009.01.006 (cit. on p. 29).

[ZLM08] Guohun Zhu, Xiangyu Luo, and Yuqing Miao. "Exact weight perfect matching of bipartite graph is NP-complete." In: *Proceedings of the World Congress on Engineering (WCE)*. 2008, pp. 1–7. URL: http://www.iaeng.org/publication/WCE2008/WCE2008_pp878-880.pdf (cit. on p. 26).