Covering a graph by nontrivial paths

Renzo Gonzalo Gómez Diaz

TESE APRESENTADA
AO
INSTITUTO DE MATEMÁTICA E ESTATÍSTICA
DA
UNIVERSIDADE DE SÃO PAULO
PARA
OBTENÇÃO DO TÍTULO
DE
DOUTOR EM CIÊNCIAS

Programa: Ciência da Computação
Orientadora: Prof.^a Dr.^a Yoshiko Wakabayashi

Durante o desenvolvimento deste trabalho o autor recebeu auxílio financeiro da CAPES

São Paulo, outubro de 2019

Covering a graph by nontrivial paths

Esta versão da tese contém as correções e alterações sugeridas pela Comissão Julgadora durante a defesa da versão original do trabalho, realizada em 02/09/2019. Uma cópia da versão original está disponível no Instituto de Matemática e Estatística da Universidade de São Paulo.

Comissão Julgadora:

- Prof.^a Dr.^a Yoshiko Wakabayashi (orientadora) - IME-USP
- Prof.^a Dr.^a Cristina Gomes Fernandes - IME-USP
- Prof.^a Dr.^a Carla Negri Lintzmayer, UFABC
- Prof. Dr. Orlando Lee, UNICAMP
- Prof. Dr. Vinicius Fernandes dos Santos, UFMG

Agradecimentos

Em primeiro lugar, gostaria de agradecer a Deus por todas as experiências que vivi, e pelas pessoas que colocou no meu caminho ao longo de todos esses anos. No Brasil recebi o sacramento da Crisma e consegui me aprofundar um pouco na minha fé. Parte importante desse processo foi a minha madrinha Maria Paula, a quem agradeço pelo carinho e preocupação pelo meu bem-estar.

Agradeço à minha família, meus pais Natalia e José, e meu irmão Alexis. *Não consigo descrever com palavras todo o amor que sinto por vocês. Fizeram um grande esforço para estar presentes no dia da minha defesa, dando-me a força que precisava para dar esse último passo. Muito obrigado por apoiar-me em todas as minhas decisões, mesmo que isso implique ficarmos separados fisicamente. De vocês, aprendi os valores e ensinamentos mais importantes, e que levo comigo cada dia.*

Desde minha chegada ao Brasil, fui acolhido pelo grupo de estudantes peruanos do Instituto de Matemática de Estatística (IME). Agradeço a Erika, Miguel, Nury, Carlos, Jainor, Ana, Hans, Lucy, pelas conversas, reuniões, e os momentos de sossego. Em especial, gostaria de agradecer a três pessoas cuja amizade foi essencial no decorrer do meu doutorado. Minha gratidão eterna ao Juan Salamanca, meu amigo desde a faculdade. Embora não seja aluno do IME, ambos viemos ao Brasil para nos tornar doutores. *Juan, agradeço teus conselhos, opiniões sinceras (mesmo sendo contrárias às minhas muitas vezes), e pela preocupação pelo meu bem-estar.* Ao Juan Gutiérrez, amigo, co-author e companheiro de piadas. *Muito obrigado pelas tardes pensando em problemas, por suportar minhas brincadeiras, enfim, pela amizade.* Finalmente, agradeço ao Frank Aguilar com quem compartilhei muitos momentos conversando, assistindo algum jogo do Peru, ou voltando para casa após passar o dia na faculdade. *Admiro muito sua dedicação e integridade.*

Ao longo da minha pós-graduação, senti-me parte de uma grande família no IME. Aos professores, pessoal administrativo, e pessoal de segurança, minha gratidão pelo apoio recebido, pelas conversas, e pela preocupação com o meu bem-estar. Agradeço aos professores Arnaldo Mandel e Cristina Fernandes pelos seus ensinamentos e paciência. Com o professor Arnaldo fiz a minha primeira disciplina de teoria na pós-graduação quando cheguei do Peru. Guardo boas lembranças das aulas de otimização combinatória. Com a professora Cristina, além de fazer duas disciplinas, tive a oportunidade de colaborar em pesquisa, o que tem sido uma valiosa experiência. Como técnico das equipes da USP em competições de programação tive a chance de compartilhar muitas seletivas e olimpíadas com o (agora Doutor) Marcio Oshiro e os professores Carlos Eduardo Ferreira e Marcel Kenji de Carli Silva. *Muito obrigado por confiar em mim para ser o técnico da USP. Também, agradeço-lhes pelos conselhos e o apoio que me deram na organização de todos os eventos*

relacionados a essas competições.

Sem dúvida, não teria conseguido culminar satisfatoriamente a pós-graduação sem a orientação da professora Yoshiko Wakabayashi. Foram muitos anos de aprendizado ao seu lado. *Agradeço sua paciência, sua confiança, e todos os seus ensinamentos e conselhos. Além de tudo o que aprendi da parte profissional, o que gostaria de destacar é o seu lado humano, preocupando-se sempre pelo meu bem-estar. Além disso, tenho uma boa lembrança das nossas charlas, todos os conselhos que você me deu ou anedotas que você contou. Eles muitas vezes me ajudaram a superar momentos complicados durante o doutorado. Minha gratidão, carinho e admiração por você são eternas.*

Quero concluir estes agradecimentos, mencionando uma experiência transformadora que aconteceu durante o meu doutorado, que como as coisas boas da vida, chegou sem eu esperar. Quando me tornei técnico das equipes da USP na Maratona de Programação, também me tornei membro do MaratonIME (agora MaratonUSP): isso mudou minha vida. Desde a minha graduação sempre fui apaixonado por esta competição, tendo acompanhado, ano a ano, os resultados da fase Latino-Americana e a Final Mundial. Por isso, quando fui convidado para ser o técnico, aceitei de imediato o desafio. Embora o MaratonUSP se renove a cada ano, gostaria de mencionar os membros com os quais iniciei esse trabalho e com quem guardo uma profunda amizade: Yan Couto, Victor Sena, Nathan Proença, Giovana Delfino, Arthur Nascimento, Victor Colombo, Luis Gustavo Bittencourt, Germano Neuenfield e Gabriel Russo. *Foram muito importantes os prêmios e os títulos que conseguimos nestes últimos anos, mas o mais valioso para mim foi essa caminhada. Vi vocês treinarem muito para conseguir seus objetivos e sonhos. Mesmo não obtendo no início os resultados que esperavam, não desistiram, e no final cada um conseguiu chegar muito longe. Ver todo esse crescimento é um dos meus maiores tesouros.* A Maratona foi e continua sendo minha paixão; espero seguir ligado a ela para sempre. Meu desejo é ver, algum dia, um grupo de jovens da América Latina conquistar uma Final Mundial.

“This idea of proving was really thrilling.”

— ***Jack Edmonds***

Resumo

GÓMEZ, R. **Cobertura de grafos por caminhos não-triviais**. 2019. 87 f. Tese (Doutorado) - Instituto de Matemática e Estatística, Universidade de São Paulo, São Paulo, 2019.

Nesta tese nosso foco é o estudo de problemas sobre cobertura por caminhos em grafos. Sejam G um grafo e $\mathcal{P}$ um conjunto de caminhos disjuntos nos vértices de G . Dizemos que $\mathcal{P}$ é uma *cobertura por caminhos* se todo vértice de G pertence a algum caminho de $\mathcal{P}$. No problema da *cobertura mínima por caminhos*, desejamos encontrar uma cobertura por caminhos cuja cardinalidade seja mínima. Neste problema, sabido ser NP-difícil, o conjunto $\mathcal{P}$ pode conter caminhos triviais. Estudamos a variante do problema onde desejamos encontrar uma cobertura por caminhos não-triviais. Inicialmente, consideramos o problema relacionado à existência de tal cobertura, e mostramos como reduzi-lo a um problema de emparelhamento. Por meio desta redução mostramos uma caracterização que nos permite encontrar, em tempo polinomial, um certificado tanto para o caso positivo quanto para o caso negativo. Uma vez que proibimos caminhos triviais, para as instâncias viáveis, podemos considerar minimizar ou maximizar o número de caminhos da cobertura. Mostramos que o problema de minimização é computacionalmente equivalente ao problema da cobertura mínima por caminhos: seus valores ótimos coincidem e têm o mesmo limiar de aproximabilidade. Além disso, propomos formulações lineares inteiras para estes problemas e apresentamos alguns resultados experimentais. No caso do problema de maximização, mostramos que este pode ser resolvido em tempo polinomial. Finalmente, também consideramos a versão com pesos do problema da cobertura por caminhos, no qual buscamos uma cobertura de peso total máximo ou mínimo (sem considerar o número de caminhos), e mostramos que o primeiro pode ser resolvido em tempo polinomial, enquanto o segundo é NP-difícil. Além disso, para o caso de maximização, mostramos um algoritmo de aproximação de fator constante. Por outro lado, quando consideramos grafos com largura arbórea limitada, mostramos que ambos os problemas podem ser resolvidos em tempo linear. Concluimos mostrando uma formulação linear inteira para o caso de cobertura de peso mínimo, e estudamos o politopo que se obtém ao relaxar a restrição de integralidade. Mostramos que existem grafos para os quais este politopo tem vértices fracionários, e exibimos algumas classes de inequações válidas (para o poliedro inteiro) que separam tais vértices.

Palavras-chave: cobertura por caminhos não-triviais, cobertura mínima, $[1, 2]$ -fator, cobertura máxima, aproximação, formulação linear inteira, cobertura com pesos, largura arbórea limitada.

Abstract

Gómez, R. **Covering graphs by nontrivial paths**. 2019. 87 f. Tese (Doutorado) - Instituto de Matemática e Estatística, Universidade de São Paulo, São Paulo, 2019.

In this thesis our aim is to study problems concerning path covers of graphs. Let G be a graph and let $\mathcal{P}$ be a set of pairwise vertex-disjoint paths in G . We say that $\mathcal{P}$ is a *path cover* if every vertex of G belongs to a path in $\mathcal{P}$. In the *minimum path cover problem*, one wishes to find a path cover of minimum cardinality. In this problem, known to be NP-hard, the set $\mathcal{P}$ may contain trivial (single-vertex) paths. We study the problem of finding a path cover composed only of non-trivial paths. First, we consider the corresponding existence problem, and show how to reduce it to a matching problem. From this reduction, we derive a characterization that allows us to find, in polynomial time, a certificate for both the YES-answer and the NO-answer. When trivial paths are forbidden, for the feasible instances, one may consider either minimizing or maximizing the number of paths in the cover. We show that the minimization problem on feasible instances is computationally equivalent to the minimum path cover problem: their optimum values coincide and they have the same approximation threshold. Moreover, we propose integer linear formulations for these problems and present some experimental results. For the maximization problem, we show that it can be solved in polynomial time. Finally, we also consider a weighted version of the path cover problem, in which we seek for a path cover with minimum or maximum total weight (the number of paths does not matter), and we show that while the first is polynomial, the second is NP-hard. Furthermore, for the maximization case, we show a constant-factor approximation algorithm. We also show that, when the input graph has bounded treewidth, both problems can be solved in linear time. To conclude, we present an integer linear formulation for the case of minimum total weight, and study the polytope obtained when the integrality constraint is relaxed. We show that there are graphs for which this polytope has fractional vertices, and we exhibit some classes of inequalities that are valid for the integral polytope and separate these vertices.

Keywords: covering by nontrivial paths, min path cover, $[1, 2]$ -factor, max path cover, approximation, integer linear formulation, weighted path cover, bounded treewidth.

Contents

List of Figures	xi
List of Tables	xiii
1 Introduction	1
2 Preliminaries	7
2.1 Graph theory	7
2.2 Linear and integer programming	9
2.3 Approximation algorithms	11
3 Covering a graph by nontrivial paths: existence and optimization	13
3.1 Known results	13
3.2 Existence of nontrivial path covers	14
3.3 Optimal nontrivial path covers	21
3.4 Final remarks	23
4 Minimum path cover as a graph parameter	25
4.1 Known results	25
4.2 Cardinality of a nontrivial path cover	27
4.3 Minimum nontrivial path cover on trees	32
4.4 Approximability of MINPC and MINNTPC	34
4.5 Integer formulations for MINPC and MINNTPC	35
4.5.1 Description of the formulations	35
4.5.2 Computational environment	37
4.5.3 Description of the experiments	37
4.5.4 Experimental results	39
4.6 Final remarks	45
5 Weighted path covers	47
5.1 Known results	47
5.2 Nontrivial path covers	48
5.3 Graphs with bounded treewidth	51

5.4	The $[1, 2]$ -factor polytope	52
5.5	Final remarks	61
6	Final considerations	63
	Bibliography	67

List of Figures

1.1	A path cover of a graph by three paths.	2
1.2	<i>a</i>) a graph G , <i>b</i>) a graph H , <i>c</i>) a path cover $\mathcal{P}$ in G ; and <i>d</i>) a mapping of G into an augmentation of H	2
2.1	<i>a</i>) a tree with bipartition given by the color classes; and <i>b</i>) a hypomatchable graph. .	8
2.2	<i>a</i>) an orientation of a cycle; and <i>b</i>) an arborescence.	9
3.1	A maximum matching that cannot be extended to a path cover in $\mathcal{P}_{1,2}^1$	16
3.2	<i>a</i>) the Edmonds-Gallai decomposition of a graph G ; <i>b</i>) a maximum matching of G . .	17
3.3	<i>a</i>) The subgraph induced by $A(G) \cup D(G)$; <i>b</i>) the graph G'	19
3.4	<i>a</i>) a nontrivial path cover of $A(G) \cup D(G)$ obtained from <i>b</i>) a matching in G'	20
4.1	Complexity status of MINPC in certain classes of perfect graphs.	26
4.2	A graph G with a minimum path cover $\mathcal{P}$ induced by the black edges and the black vertices.	27
4.3	<i>a</i>) Modifying $\mathcal{P}$ using an edge linking a trivial path to a long path; and <i>b</i>) the resulting path cover.	27
4.4	<i>a</i>) Modifying $\mathcal{P}$ using an alternating path from a trivial path to a long path; and <i>b</i>) the resulting path cover.	28
4.5	<i>a</i>) Modifying $\mathcal{P}$; and <i>b</i>) the resulting path cover.	28
4.6	(<i>a</i>) a graph G and $\mathcal{P}$, and (<i>b</i>) the $\mathcal{P}$ -digraph D	29
4.7	(<i>a</i>) a path P ; and (<i>b</i>) the resulting path cover $\mathcal{Q}$	31
4.8	Running time of TP around the Hamiltonicity threshold.	40
4.9	Running time of NTP around the Hamiltonicity threshold.	40
4.10	Running time of TP on dense graphs.	42
4.11	Running time of NTP on dense graphs.	42
4.12	Running time for TP in graphs that obey the Barabási & Albert model.	44
4.13	Running time for NTP in graphs that obey the Barabási & Albert model.	44
5.1	The T -reduction of a K_3	49
5.2	<i>a</i>) a graph G ; and <i>b</i>) a vertex x^1 of $P(G)$	54
5.3	<i>a</i>) a graph G and <i>b</i>) a vertex x^2 of $P(G)$ separated by inequality (5.5).	56

5.4	<i>a)</i> K_4 and <i>b)</i> a vertex of $P(K_4)$	57
5.5	<i>a)</i> a graph G ; and <i>b)</i> a fractional vertex of $P(G)$	58

List of Tables

4.1	Polynomial-time algorithms for MINPC in certain classes of perfect graphs.	26
4.2	Computational environment.	37
4.3	Hamiltonicity threshold $p^*(n)$ for some values of n	38
4.4	Results obtained by TP.	39
4.5	Results obtained by NTP.	40
4.6	Number of instances containing a nontrivial path cover in the Erdős & Rényi model.	41
4.7	Results obtained for $p = 30\%$	41
4.8	Results obtained for $p = 50\%$	41
4.9	Results obtained for $p = 70\%$	43
4.10	Results obtained for TP.	43
4.11	Results obtained for NTP.	43
4.12	Number of instances containing a nontrivial path cover in the Barabási & Albert model.	44
5.1	Approximation algorithms for MAXPC.	48
6.1	Computational complexity and approximability of the problems we studied.	63

Chapter 1

Introduction

A classical and widely studied problem in graph theory is the Hamiltonian path problem. In this problem, we want to decide whether a given graph has a path containing all of its vertices. It is known to be NP-complete for many classes of graphs, such as circle graphs [Dam89], planar 3-connected graphs [GJ79], split graphs [Gol04], chordal bipartite graphs [M96], directed path graphs [Nar89], etc. This decision problem is naturally related to some combinatorial optimization problems. For example, if instead of asking whether a graph G has a Hamiltonian path, we ask for the length of the longest path in G , we would solve the former problem indirectly. Another related problem is the following one.

Given a graph G , we may look for a covering of $V(G)$ by vertex-disjoint paths. Observe that, if we seek such a cover with minimum size, then we would also solve the Hamiltonian path problem indirectly. The latter problem motivated our studies and the results presented in this text. We define it formally as follows.

MINIMUM PATH COVER (MINPC): Given a graph G , find a collection $\mathcal{P}$ of vertex-disjoint paths that covers $V(G)$ and minimizes $|\mathcal{P}|$.

In Figure 1.1 we show a path cover of a graph by three paths, two of them induced by the wavy edges, and the third one represented by the shaded vertex. The first work on the MINPC problem appeared in the literature in 1974. Boesch et al. [BCM74] studied the cardinality of a minimum path cover as a graph parameter. They showed a closed formula for this parameter in the case of complete k -ary trees and; in the case of general graphs, they showed upper bounds depending on the degree sequence of the graph. Three years later, Boesch & Gimpel [BG77] motivated by a problem in code optimization, studied the MINPC problem in directed graphs. They showed a polynomial-time algorithm for MINPC on directed acyclic graphs by reducing it to a maximum matching problem in a bipartite graph. Moreover, this is the first time the equivalence between a path cover with minimum number of paths and one with maximum number of edges is mentioned in the literature. Some years later, in 1987, Pinter & Wolfstahl [PW87] studied a problem about mapping processes to processors in parallel architectures, which we state as follows.

Figure 1.1: *A path cover of a graph by three paths.*

A parallel program is a set of processes that we model as a graph $G = (V, E)$ where the vertices represent the processes and an edge between two vertices, say u and v , indicates that the processes represented by u and v interact during the execution of the program. On the other hand, a network of processors is a set of computers that are linked physically. We model such network as a graph $H = (V', E')$ where the processors are represented by the set V' and an edge in E' represents the existence of a physical link. The problem Pinter & Wolfstahl addressed is an augmentation problem, where we wish to add a minimum number of edges to H so that the resulting graph contains a subgraph isomorphic to G . For application reasons, they considered the special case where H is a (Hamiltonian) path. We show in Figure 1.2 an example of such augmentation problem.

Figure 1.2: *a) a graph G , b) a graph H , c) a path cover $\mathcal{P}$ in G ; and d) a mapping of G into an augmentation of H .*

The edges to be added to H indicate the links that must be added to the network of processors to run the parallel program (represented by G). To show the relation between this augmentation problem and MINPC, consider a graph G with n vertices and m edges. Without loss of generality, suppose that $|V(H)| = |V(G)|$. Since H is a path, for any mapping of $V(G)$ into $V(H)$, the set of edges of G that are contained in H (under that mapping) together with non-covered vertices define a path cover of G , say $\mathcal{P}$. As $n - |\mathcal{P}|$ edges of G are contained in H , we have that

$$m = n - |\mathcal{P}| + |A|,$$

where A is the set of edges that we need to add to H in order to obtain a subgraph isomorphic to G . This equality shows that minimizing $|A|$ is equivalent to minimizing $|\mathcal{P}|$. In Figure 1.2 c) we show a minimum path cover by three paths: one induced by the bold edges, and the other two represented by the shaded vertices. In Figure 1.2 d), we show an optimal mapping where the edges in A are represented by the dotted edges.

Pinter & Wolfstahl also showed a linear-time algorithm to solve the MINPC problem on a subclass of cactus graphs. Motivated by their work, many researchers devised polynomial-time algorithms for the MINPC problem in various subclasses of perfect graphs such as bipartite permutation graphs [SSSP93], interval graphs [AP90], cocomparability graphs [CDH13], circular-arc graphs [HC06], etc.

As we mentioned before, the first results on MINPC regarded the cardinality of a path cover as a graph parameter. In this case, we are interested in showing upper bounds for this number as well as graphs that attain that bound (if possible). After the results shown by Boesch et al. [BCM74] many years passed with no further discoveries in this direction. In 1996, Reed [Ree96] studied this parameter for the class of cubic graphs. He showed that $\lceil \frac{n}{9} \rceil$ paths are enough to cover any cubic graph of order n . Moreover, he exhibited an infinite family of cubic graphs that attain this bound. It is worth mentioning that this is the only result that shows a tight upper bound for a considerably large class of graphs. He also conjectured that $\lceil \frac{n}{10} \rceil$ paths is a tight upper bound for the class of 2-connected cubic graphs. In 2009, Magnant & Martin [MM09] showed an upper bound for this parameter in regular graphs under specific conditions. More recently, Magnant et al. [MWY16] showed some upper bounds for general graphs depending on the maximum and minimum degree of the graph. In 2017, Yu [Yu18], motivated by the conjecture of Reed [Ree96], showed an upper bound of $\lceil \frac{n}{10} \rceil$ for the minimum path cover number in cubic 2-connected graphs. Although this is the same bound conjectured by Reed, Yu showed that the example given by Reed was not correct. Moreover, Yu exhibited an infinite family of 2-connected cubic graphs that need at least $\lceil \frac{n}{14} \rceil$ paths in any path cover. It is still an open question to settle whether there exists a 2-connected cubic graph that requires $\lceil \frac{n}{10} \rceil$ paths to cover it, or to show that $\lceil \frac{n}{14} \rceil$ paths are enough to cover any 2-connected cubic graph.

Finally, the minimum path cover of a graph is also related to other graph parameters. In 1994, Georges et al. [GMW94] showed a relation between a minimum $L(2, 1)$ -labelling (a special type of vertex-coloring) of a graph G and a minimum path cover of $\overline{G}$, the complement of G . They proved

that if $\overline{G}$ does not have a Hamiltonian path, then the minimum number of colors in an $L(2, 1)$ -labelling of G is $|V(G)| + |\mathcal{P}| - 2$, where $\mathcal{P}$ is a minimum path cover of $\overline{G}$. Their proof also shows how, given a minimum path cover of $\overline{G}$, we can obtain the desired coloring. Recently, Henning & Wash [HW17] showed that the minimum path cover number of a graph G along with the size of a maximum matching gives an upper bound on the total domination number of G . As we have seen, the minimum path cover problem is not only of practical interest but also of theoretical interest, as it is related to many classical problems in graph theory.

In this thesis, our main focus is to study a variant of the path cover problem that arises from the following question: “*what happens when we prohibit trivial (single-vertex) paths in the path cover?*” Observe that, in this case, deciding the existence of such path cover is a problem in its own right. Moreover, after dealing with its existence, we may either want to minimize or maximize the number of paths in the cover. We devoted great part of our studies to these optimization problems, which we denote and define formally as follows

MINIMUM NONTRIVIAL PATH COVER (MINNTPC): Given a graph G , find a non-trivial path cover $\mathcal{P}$ such that $|\mathcal{P}|$ is minimum, or answer that none exists.

MAXIMUM NONTRIVIAL PATH COVER (MAXNTPC): Given a graph G , find a non-trivial path cover $\mathcal{P}$ such that $|\mathcal{P}|$ is maximum, or answer that none exists.

In what follows we give an overview of the organization of this text. First, in Chapter 2 we introduce some basic definitions and the terminology we use throughout this text. After that, in Chapter 3 we focus on the existence of a nontrivial path cover of a graph. We characterize the class of graphs that admit this kind of path cover. From this characterization, we derive a polynomial-time algorithm that either finds a nontrivial path cover or a set of vertices that represents a negative certificate for the existence of such cover. After coping with the existence problem, we focus on MINNTPC and MAXNTPC. We conclude this chapter showing that MAXNTPC can be solved in polynomial time. In fact, our algorithm to solve the existence problem returns an optimal solution for MAXNTPC in case of a YES-answer.

In Chapter 4 we focus on MINPC and MINNTPC. We show that, when a nontrivial path cover exists, the cardinality of an optimal solution to MINNTPC is the same as the cardinality of an optimal solution to MINPC. Indeed, we show a polynomial-time algorithm that transforms any path cover $\mathcal{P}$ of a graph into one without trivial paths and with at most $|\mathcal{P}|$ paths. This result has important implications for the computational complexity and approximability of MINNTPC. After that, we show that MINNTPC can be solved in linear-time for trees, and derive some approximability results for MINPC and MINNTPC. We conclude this chapter showing two integer linear programming formulations, one for MINPC and the other for MINNTPC, along with experimental results for these formulations.

In Chapter 5 we consider edge-weighted graphs. In this case we are interested in path covers of maximum (minimum) total weight (the number of paths does not matter). First, we present the results known for these problems in the literature. After that, we focus on weighted nontrivial path covers. We call these problems MINIMUM WEIGHT NONTRIVIAL PATH COVER (MINWNTPC) and MAXIMUM WEIGHT NONTRIVIAL PATH COVER (MAXWNTPC). We observe that on unit-weight graphs MAXWNTPC is equivalent to MINNTPC. That is, a path cover of a graph with maximum number of edges also has the minimum number of paths possible. Conversely, MINWNTPC is equivalent to MAXNTPC on the same class of weighted graphs. As in the case of MAXNTPC, we show that MINWNTPC can be solved in polynomial time. On the other hand, we show that MAXWNTPC is NP-hard, and propose a $(2/3)$ -approximation algorithm for this problem. We conclude this chapter with a study of the polytope defined by the incidence vectors of the $[1, 2]$ -factors of a graph, which is a concept that is closely related to the nontrivial path cover. Finally, in Chapter 6 we present some concluding remarks on the problems studied in this thesis.

Chapter 2

Preliminaries

In this chapter we introduce the basic concepts, notation and terminology we use throughout this thesis. Although the notation and terminology we use are conventional in graph theory, polyhedral combinatorics, and approximation algorithms, they are included to make the text self-contained. The terminology we adopt for graphs follows Bondy & Murty [BM08] and Diestel [Die18]. The reader may refer to Schrijver [Sch03] for the basic concepts and definitions in polyhedral combinatorics, and for topics on approximation algorithms [CCD⁺01].

2.1 Graph theory

A **graph** is a pair $G = (V, E)$, where $V \cap E = \emptyset$, along with a function ψ that maps each element in E to a subset of V of cardinality 2. The elements of V and E are called **vertices** and **edges**, respectively. Given a graph G , we denote by $V(G)$ and $E(G)$ its set of vertices and its the set of edges, respectively. The **order** of a graph G is the cardinality of $V(G)$. If $e \in E$, and $\psi(e) = \{u, v\}$, where possibly $u = v$, we say that e **links** the vertices u and v , and that u and v are the **ends** of e . Moreover, we say that e is **incident** to u or v , and that u and v are **neighbors** or **adjacent**. In case $u = v$, we say that e is a **loop**, and if there exist $a, e \in E$ such that $\psi(a) = \psi(e)$, then we say that a and e are **parallel**. We say that a graph is **simple** if it contains neither loops nor parallel edges. Note that, if G is a simple graph, then we can define the set of edges in G by specifying which are the subsets of distinct vertices in $E(G)$, leaving the function ψ implicitly defined. In this case, if $\psi(e) = \{u, v\}$, we abbreviate this fact by writing $e = uv$.

If $G' = (V', E')$ is a graph such that $V' \subseteq V$ and $E' \subseteq E$, then, we say that G' is a **subgraph** of G and that G is a **supergraph** of G' . Given $S \subseteq V$, the **subgraph of G induced by S** is the subgraph $\mathbf{G}[S] := (S, A)$, where $A \subseteq E$ is the subset of edges of E whose ends belong to S . Furthermore, we denote by $\mathbf{G} - \mathbf{S}$ the subgraph $G[V \setminus S]$, and if $F \subseteq E$, we denote by $\mathbf{G} - \mathbf{F}$ the graph $(V, E \setminus F)$. If α is a vertex or an edge of G , we write $\mathbf{G} - \alpha$ instead of $G - \{\alpha\}$. Let $v \in V$. The **neighborhood** of a vertex v in G is $\mathbf{N}(v) := \{u \in V : u \text{ is adjacent to } v\}$. Let $S \subseteq V$. The **cut** induced by S is the set of edges with one end in S and the other in $V \setminus S$. We denote it by $\delta(S)$. When $S = \{v\}$, for any $v \in V$, we write $\delta(v)$ instead of $\delta(\{v\})$. The **degree** of v in G ,

denoted by $\deg_G(v)$ or simply $\deg(v)$, is $|N(v)|$. We denote by $\delta(G)$ and $\Delta(G)$ the **minimum degree** and the **maximum degree** of G , respectively; that is $\delta(G) := \min\{\deg(v) : v \in V\}$ and $\Delta(G) := \max\{\deg(v) : v \in V\}$.

A **path** P , in G , is a sequence $P := \langle v_1, e_1, v_2, \dots, e_k, v_{k+1} \rangle$, where $v_1, \dots, v_{k+1}$ are distinct vertices of G and e_i is an edge that links v_i and v_{i+1} for $i = 1, 2, \dots, k$. In this case, we say that P is a path between v_1 and v_{k+1} , or that P links v_1 and v_{k+1} , and that v_1 and v_{k+1} are the **ends** of P . A vertex that is called **internal** vertex of P is not an end of P . The **length** of a path P , denoted $|P|$, is the number of its edges. We say that a path is **trivial** if its length is zero, and we call a path a **k -path** if its length is k . Usually, we represent a path P just by its sequence of vertices, writing succinctly $P = \langle v_1, \dots, v_{k+1} \rangle$. We say that G is **connected** if there exists a path in G that links vertices u and v , for any pair $u, v \in V$. A **component** of G is a maximal connected subgraph of G . The **distance** between a pair of vertices u, v in a connected graph G is the length of a shortest path that links u and v in G . A **cycle** C , in G , is a sequence $C := \langle v_1, e_1, v_2, \dots, v_k, e_k, v_{k+1} \rangle$, where $v_1, v_2, \dots, v_k$ are distinct vertices of G , $v_1 = v_{k+1}$ and e_i links v_i to v_{i+1} , for $i = 1, 2, \dots, k$. If G does not contain a cycle, we say that G is **acyclic**. An acyclic graph is also called a **forest**. A **tree** is a connected forest. A vertex of degree one in a tree is called a **leaf**.

Figure 2.1: a) a tree with bipartition given by the color classes; and b) a hypomatchable graph.

We say that G is a **bipartite** graph if there exist nonempty subsets, say X and Y , of $V(G)$ such that X and Y are disjoint, $X \cup Y = V(G)$ and every edge in $E(G)$ links a vertex in X to a vertex in Y . In this case, we also say that G is (X, Y) -**bipartite**. If G is an (X, Y) -bipartite graph such that $xy \in E(G)$, for every pair of vertices $x \in X$ and $y \in Y$, we say that G is a **complete bipartite** graph. We denote by $K_{m,n}$ the complete (X, Y) -bipartite graph, where $|X| = m$ and $|Y| = n$. Note that there exists a unique $K_{m,n}$ graph (up to isomorphism).

In a graph $G = (V, E)$, a subset $M \subseteq E$ is a **matching** if M does not contain loops and there is at most one edge of M incident to any vertex v in $V(G)$. Let M be a matching of a graph G . We say that a vertex $v \in V(G)$ is **covered by M** , or that it is **M -matched**, if there is an edge of M that is incident to v ; and if every vertex of G is covered by M , we say that M is **perfect**. We say that a path P is **M -alternating** if the edges of P belong alternately to M and $E(G) \setminus M$, and we say that it is an **M -augmenting** path if both of its ends are not covered by M . A graph is **hypomatchable** if $G - v$ contains a perfect matching for every v in G (see Figure 2.1 b)).

A **path cover** of a graph G is a collection $\mathcal{P} = \{P_1, \dots, P_k\}$ of vertex-disjoint paths in G such that $\cup_{i=1}^k V(P_i) = V(G)$.

A **directed graph**, or *digraph*, is a pair $D = (V, A)$, where $V \cap A = \emptyset$, along with a function ψ

Figure 2.2: a) an orientation of a cycle; and b) an arborescence.

that associates an ordered pair of elements (not necessarily distinct) of V to each element of A . The elements of V and A are called **vertices** and **arcs**, respectively. We denote by $V(D)$ and $A(D)$ the set of vertices and the set of arcs, respectively, of a directed graph D . If $a \in A$, and $\psi(a) = (u, v)$, we say that u is the **tail** of a and that v is its **head**. We also say that a **begins** at u and **ends** at v .

For $S \subseteq V$, the **subgraph of D induced by S** is the subgraph $D[S] := (S, F)$, where $F \subseteq A$ is the subset of arcs in A that begin and end at a vertex of S . Moreover, we denote by $D - S$ the subgraph $D[V \setminus S]$, and if $F \subseteq A$, we denote by $D - F$ the directed graph $(V, A \setminus F)$. If α is a vertex or an arc of D , we write $D - \alpha$ instead of $D - \{\alpha\}$. Let $v \in V(D)$. The **outdegree** of v in D , denoted by $\deg_D^+(v)$ or simply $\deg^+(v)$, is the number of arcs that begin at v . The **indegree** of v in D , denoted by $\deg_D^-(v)$ or simply $\deg^-(v)$, is the number of arcs that end at v .

A **path** P , in D , is a sequence $P := \langle v_1, a_1, v_2, \dots, a_k, v_{k+1} \rangle$, where $v_1, \dots, v_{k+1}$ are distinct vertices of D and a_i is an arc with tail v_i and head v_{i+1} , for $i = 1, 2, \dots, k$. In this case, we say that P is a path that **begins** in v_1 and **ends** in v_{k+1} , and has **length** k . Most of the time, we represent a path P just by its sequence of vertices, writing $P = \langle v_1, \dots, v_{k+1} \rangle$.

An **orientation** of a graph $G = (V, E)$ is the digraph that results from G when we replace each edge with one of the two arcs that have the same ends (see Figure 2.2 a)). A **rooted tree** T is an orientation of a tree in which every vertex but one has indegree one. The vertex with indegree zero is called the **root** of T . We also refer to a rooted tree with root r as an **r -arborescence**. In Figure 2.2 b), we show an example of an arborescence.

For the readers interested in these and other concepts in graph theory, we suggest the books by Bondy & Murty [BM08] and Diestel [Die18].

2.2 Linear and integer programming

In **linear programming**, given vectors $c \in \mathbb{R}^n$, $b \in \mathbb{R}^m$ and a matrix $A \in \mathbb{R}^{m \times n}$, we want to find a vector $x \in \mathbb{R}^n$ that

$$\begin{aligned} & \text{minimize} && \sum_{i=1}^n c_i x_i \\ & \text{subject to} && \sum_{j=1}^n A_{ij} x_j \leq b_i, \quad \text{for } i = 1, \dots, m. \end{aligned} \tag{2.1}$$

Each inequality $\sum_{j=1}^n A_{ij}x_j \leq b_i$ is called a **constraint** of (2.1). We say that c is the **objective function** of (2.1). A vector $x \in \mathbb{R}^n$ satisfying the constraints in (2.1) is called a **feasible solution**. Moreover, if

$$\sum_{i=1}^n c_i x_i \leq \sum_{i=1}^n c_i x'_i,$$

for any feasible solution x' , we say that x is an **optimal solution** of (2.1). In the model presented above we may also consider lower bound inequalities, equalities or maximizing the objective function. It can be proved that all these variants can be modeled by a linear program as above. We can solve linear programming in polynomial time, either by the Ellipsoid Method [Kha79] or the Interior Point Method [Kar84]. In practical applications, the Simplex method is the most used algorithm to solve linear programs by its practical efficiency.

In **integer programming**, besides the constraints given by the inequalities, the vector x must belong to $\mathbb{Z}^n$. It is a known fact that, in general, solving integer programming is NP-hard [GJ79]. The importance of integer programming lies in its power to model many problems in combinatorial optimization. When we want to solve a specific integer program, say (I) , one common step is to drop the integrality constraint over its variables and study what is called the **linear relaxation** of (I) . Observe that this linear program already gives a lower bound on the value of an optimal solution of (I) . To further improve this bound or solve (I) in polynomial time, researchers study the properties of the points that correspond to feasible solutions of the linear relaxation of (I) . This approach is part of what is known as **polyhedral combinatorics**.

Let x be a vector in $\mathbb{R}^n$. The **components** of $x = (x^1, \dots, x^n)$ are $x^1, \dots, x^n$. The **support** of x is the subset $I \subseteq \{1, \dots, n\}$ such that $x^i \neq 0$, for $i \in I$. Let $G = (V, E)$ be a graph and let F be a subset of E . The **incidence vector** of F , denoted by χ^F , is the vector in $\mathbb{R}^E$ defined as

$$\chi^F(e) := \begin{cases} 1, & \text{if } e \in F, \\ 0, & \text{otherwise,} \end{cases}$$

for $e \in E$. Let $x_1, \dots, x_m$ be vectors in $\mathbb{R}^n$. We say that the vector $x \in \mathbb{R}^n$ is a **linear combination** of $x_1, x_2, \dots, x_m$ if there exist real numbers $\lambda_1, \dots, \lambda_m$ such that $x = \sum_{i=1}^m \lambda_i x_i$. Moreover, a linear combination such that $\sum_{i=1}^m \lambda_i = 1$ and $\lambda_i \geq 0$, for $i = 1, 2, \dots, m$, is called **convex**. A convex combination $\sum_{i=1}^m \lambda_i x_i$ is called **proper** if $\lambda_i > 0$, for $i = 1, \dots, m$. The vectors $x_1, x_2, \dots, x_m$ are **linearly independent** if $\sum_{i=1}^m \lambda_i x_i = \vec{0}$ implies that $\lambda_i = 0$, for $i = 1, \dots, m$. The **convex hull** of a set of vectors $S \subseteq \mathbb{R}^n$ is the set of all convex combinations of elements in S . A **polyhedron** P in $\mathbb{R}^n$ is a set of points that can be described as $\{x \in \mathbb{R}^n : Ax \leq b\}$ where A is an $m \times n$ matrix and b is a vector in $\mathbb{R}^m$. A **hyperplane** is a polyhedron of the form $\{x \in \mathbb{R}^n : cx = \delta\}$ for some $c \in \mathbb{R}^n$ and $\delta \in \mathbb{R}$. We say that a polyhedron P is a **polytope** if it is the convex hull of a finite set of points. A point x is called a **vertex** of a polytope $P \subseteq \mathbb{R}^n$ if it is not a proper convex combination of two distinct points in P . The vertices of a polytope can also be characterized in another useful way. A point $x' \in P$ is a vertex if there exists a hyperplane H such that

$$\{x'\} = P \cap H.$$

Most of the time, in order to obtain a complete description of a polytope arising from a linear relaxation, we need to consider an exponential number of constraints in our formulation. A seminal result by Grötschel, Lovász, and Schrijver [GLS93] helps us cope with this difficulty. Let $P \subset \mathbb{R}^n$ be a polytope. The **separation problem** for P is the problem where, given a vector $z \in \mathbb{R}^n$, we want to find a vector $c \in \mathbb{R}^n$ such that

$$\sum_{i=1}^n c_i x_i > \sum_{i=1}^n c_i z_i$$

for every $x \in P$, if such c exists. Observe that, if $z \notin P$, the vector c gives us a separating hyperplane. Grötschel, Lovász, and Schrijver proved that, under certain assumptions over the size of the input polytope, the problem of finding an optimal solution for P is polynomial-time equivalent to the separation problem for P . Moreover, they showed how to solve a linear program with an exponential number of constraints in polynomial time using the Ellipsoid Method and a polynomial-time algorithm (oracle) that solves the separation problem for the polytope described by the linear program.

For the readers interested in linear programming we suggest the book by Chvátal [Chv83]. In the case of integer programming and polyhedral combinatorics we suggest the books by Conforti et al. [CCZ14] and Schrijver [Sch03].

2.3 Approximation algorithms

An optimization problem can be described by a triple $(\mathcal{I}, \mathcal{S}, f)$, where

- $\mathcal{I}$ is the set of instances,
- $\mathcal{S}$ is the set of feasible solutions, and
- $f : \mathcal{S} \rightarrow \mathbb{R}$ is a function that assigns the value $f(S)$ to a solution $S \in \mathcal{S}$.

In optimization problems we are interested in either minimizing or maximizing the value of a feasible solution. When one of these alternatives is implicitly understood, we say that this solution (or its value) is **optimum**. We denote by $\text{OPT}(I)$ the value of an optimum solution for the instance I .

Let $\Pi = (\mathcal{I}, \mathcal{S}, f)$ be a minimization problem. Let $\mathcal{A}$ be an algorithm that, for each $I \in \mathcal{I}$, returns a solution $\mathcal{A}(I) \in \mathcal{S}$ for I . Let α be such that $\alpha \geq 1$, where α is a constant or a function of I . We say that $\mathcal{A}$ is an α -**approximation** for Π if it runs in polynomial time on the size of the representation of I , and

$$f(\mathcal{A}(I)) \leq \alpha \text{OPT}(I), \text{ for every } I \in \mathcal{I}.$$

In case Π is a maximization problem, $\alpha < 1$ and

$$f(\mathcal{A}(I)) \geq \alpha \text{OPT}(I), \text{ for every } I \in \mathcal{I}.$$

We also say that α is the **approximation factor** of $\mathcal{A}$. We observe that the value α may depend on the size of the instance I , and in this case, instead of α , we write $\alpha(I)$. For example, if I is an instance that is a graph of order n , then $\alpha(I)$ could be functions such as $\log(n)$ or $\sqrt{n}$.

A **Polynomial-Time Approximation Scheme** (PTAS) for Π is a family of algorithms $\{\mathcal{A}_\epsilon\}$, where $\mathcal{A}_\epsilon$ is a $(1 + \epsilon)$ -approximation for each $\epsilon > 0$. If Π is a maximization problem, then $\mathcal{A}_\epsilon$ is a $(1 - \epsilon)$ -approximation algorithm. Observe that a PTAS can be seen as an algorithm that receives as input an instance I and an $\epsilon > 0$. This algorithm runs in polynomial time on the size of the representation of I where we regard $1/\epsilon$ as constant. If this algorithm also runs in polynomial time in $1/\epsilon$, then we say that $\{\mathcal{A}_\epsilon\}$ is a **Fully Polynomial-Time Approximation Scheme** (FPTAS).

An **approximation threshold** for Π is the maximum lower bound for the values of α such that there is an α -approximation for Π . In other words, Π does not admit an $(\alpha - \epsilon)$ -approximation for any $\epsilon > 0$. For maximization problems, we exchange maximum lower bound by minimum upper bound in the previous definition.

For the readers interested in more concepts and results in approximation algorithms, we suggest the books by Carvalho et al. [CCD⁺01], Williamson & Shmoys [WS11], and Vazirani [Vaz01].

Chapter 3

Covering a graph by nontrivial paths: existence and optimization

This chapter is primarily devoted to the study of graphs that admit a nontrivial path cover. In Section 3.1, we present the results and characterizations that are known in the literature on this topic. Also, we introduce the concept of a $[1, 2]$ -factor of a graph which is intimately related to the concept of a nontrivial path cover in a graph. After that, in Section 3.2 we show a relation between the existence of a nontrivial path cover and the maximum matchings of a graph. This relation allows us to prove a new characterization of graphs that admit such path cover. Finally, in Section 3.3 we focus on MINNTPC and MAXNTPC. We show that MAXNTPC can be solved in polynomial time, while MINNTPC is NP-hard.

3.1 Known results

A first question on the problem of covering a graph by nontrivial paths is whether every graph admits such a cover. A moment's thought leads us to a negative answer. For example, any star with at least four vertices does not have such a path cover. To exclude such examples, one may naturally ask whether minimum degree two would suffice, but this is not the case. To show this, consider the graph $K_{2,5}$. Since in every nontrivial path cover of $K_{2,5}$ the vertices of degree five can be adjacent to at most two vertices of the graph, then one of the vertices of degree two cannot be covered. Indeed, as noted by Kano & Saito [KS83], every $K_{r,r+s}$ with $s > r \geq 1$ does not admit a nontrivial path cover. This also shows that, for every positive integer k , there exists a graph G with $\delta(G) \geq k$ that cannot be covered by nontrivial paths.

In fact, the nontrivial path cover existence problem is a special case of a classical and intensively studied problem in graph theory. An $[a, b]$ -factor of a graph G is a spanning subgraph H of G such that every vertex in H has degree at least a and at most b , where a and b are constants. Observe that a nontrivial path cover is a $[1, 2]$ -factor of a graph. Moreover, we can remove one edge from every cycle in a $[1, 2]$ -factor to obtain a nontrivial path cover of a graph. Thus, asking for a $[1, 2]$ -factor of a graph is equivalent to asking for a nontrivial path cover. A natural generalization of

the concept of $[a, b]$ -factor is the concept of (g, f) -factor, in which we replace the constants a and b by integer functions g and f . In what follows, we summarize the most important results regarding $[1, 2]$ -factors and (g, f) -factors.

In 1983, Kano & Saito [KS83] proved that for $r \geq 1$, if G is a graph such that $\delta(G) \geq r$ and $\Delta(G) \leq r + s$, where $s \in \{1, \dots, r\}$, then G contains a $[1, 2]$ -factor. In 1998, Li & Mao-Cheng [LM98] showed a degree condition that, when specialized to $[1, 2]$ -factors, guarantees that: if G is a graph of order $n \geq 4$ such that $\max\{\deg(x), \deg(y)\} \geq n/3$, for all non-adjacent vertices x and y , then G has a $[1, 2]$ -factor. Regarding (g, f) -factors, the two most important characterizations are those obtained by Tutte and Lovász. In 1952, Tutte [Tut52] characterized the graphs that have an (f, f) -factor. In 1970, Lovász [Lov70] generalized this result characterizing graphs that have a (g, f) -factor. This result when specialized to $[1, 2]$ -factors will be of great interest for us, so we state it below (see Las Vergnas [LV78]).

Theorem 3.1 (Lovász, 1970). *A graph G has a $[1, 2]$ -factor if, and only if, $i(G - S) \leq 2|S|$ for every $S \subseteq V(G)$, where $i(G - S)$ is the number of isolated vertices in $G - S$.*

Regarding the algorithmic aspect of this problem, some results were also obtained. Given a graph of order n , Anstee [Ans85] showed an $\mathcal{O}(n^3)$ algorithm that finds a (g, f) -factor, if one exists, or a (g, f) -barrier (which is a negative certificate). In 1990, Heinrich et al. [HHKL90] showed a faster algorithm that finds a (g, f) -factor when $g(v) \leq 1$ and $g(v) < f(v)$ for every vertex v in G . These results already tell us that the nontrivial path cover existence problem can be solved in polynomial time.

We observe that most of these algorithms, except for Anstee [Ans85], did not find a negative certificate (a set S that does not satisfy the condition stated in Theorem 3.1). In the next section we describe our approach to solve the existence problem. We show that this problem is closely related to the maximum matchings of a graph. Moreover, the proof of our characterization implies a polynomial-time algorithm that, in the case of a negative answer, returns a set of vertices which also violates the condition given in Theorem 3.1.

3.2 Existence of nontrivial path covers

Let G be a graph. Clearly, if G has a perfect matching, then it has a path cover consisting solely of 1-paths. Note that, if G admits a nontrivial path cover, but it does not have a perfect matching, then we need at least one k -path, with $k > 1$, to cover it. In fact, we do not have to consider $k > 2$, since paths with length greater than two can be broken into paths of length one or two. This observation indicates that we just need to focus on the problem of deciding whether a graph G admits a nontrivial path cover consisting solely of 1-paths and 2-paths. This special kind of path covers is the crux of our approach to decide the existence of a nontrivial path cover in a graph.

We denote by $\mathcal{P}_{1,2}(G)$, or simply $\mathcal{P}_{1,2}$, this class of nontrivial path covers of G . Inside this class, we will be specially interested in the subclass $\mathcal{P}_{1,2}^1(G)$ consisting of the *path covers in $\mathcal{P}_{1,2}$ with*

the largest possible number of 1-paths. The following proposition relates the path covers in $\mathcal{P}_{1,2}^1(G)$ to the maximum matchings of G .

Proposition 3.2. *If a graph G admits a nontrivial path cover, then the cardinality of any path cover in $\mathcal{P}_{1,2}^1(G)$ coincides with the cardinality of a maximum matching in G .*

Proof. Consider a path cover $\mathcal{P}' \in \mathcal{P}_{1,2}^1(G)$, and let $\mathcal{P}'_2$ be the set of 2-paths in $\mathcal{P}'$. Let M' be a matching of G obtained by choosing one edge from every path in $\mathcal{P}'$. We claim that M' is a maximum matching of G . Suppose, on the contrary, that this claim does not hold. Let P be an M' -augmenting path that intersects a minimum number of paths in $\mathcal{P}'_2$, and let u and v be its ends. Let P_u and P_v be the paths in $\mathcal{P}'_2$ whose ends are u and v , respectively. Next, we show that P_u and P_v are the unique paths in $\mathcal{P}'_2$ that intersect P .

Suppose, by contradiction, that there exists a path $Q \in \mathcal{P}'_2$, distinct from P_u and P_v , that intersects P . Consider that $Q := \langle x, y, z \rangle$, and that z is the end of Q not covered by M' . (The proof for the case x is the end of Q not covered by M' is analogous.) Since every internal vertex of P is incident to an edge in M' , it follows that $z \notin V(P)$. Therefore, x and y are adjacent vertices in P . Suppose first that $P = \langle u, \dots, x, y, w, \dots, v \rangle$. In this case, since $xy \in M'$, then $yw \notin M'$. Moreover $w \neq z$, so the path $\langle u, \dots, x, y, z \rangle$ is an M' -augmenting path that contradicts the choice of P . If $P = \langle u, \dots, y, x, w, \dots, v \rangle$, then the path $\langle z, y, x, \dots, v \rangle$ is an M' -augmenting path that contradicts the choice of P .

Since P is an M' -augmenting path from u to v , then P_u and P_v are subpaths (the initial and final part) of P . Now, to conclude the proof, let $E(\mathcal{P}')$ be the set of edges of the path cover $\mathcal{P}'$. Then, the edges in $(E(\mathcal{P}') \setminus (M' \cap E(P))) \cup (E(P) \setminus M')$ define a path cover of G by 1-paths and 2-paths, containing more 1-paths than $\mathcal{P}'$, a contradiction. This concludes the proof that M' is a maximum matching of G , as we claimed. $\square$

Proposition 3.2 tells us that, if a graph G admits a nontrivial path cover, then any path cover in $\mathcal{P}_{1,2}^1(G)$ contains a maximum matching of G . This may tempt us to think that, given any maximum matching of G , we can extend it (by adding some edges) to obtain a path cover in $\mathcal{P}_{1,2}^1(G)$. Indeed, this can be done for regular graphs. Let G be a k -regular graph, $k \geq 1$, and let M be a maximum matching of G . Let X (resp. Y) be the M -unmatched (resp. M -matched) vertices of G . Consider H as the (X, Y) -bipartite graph induced by the edges of G with one end in X and the other in Y . By the maximality of M , we have that $\deg_H(x) = \deg_G(x) = k$, for every $x \in X$. This implies that $\deg_H(y) \leq \deg_H(x)$, for every $x \in X$ and $y \in Y$. So, by Hall's Theorem, there is a matching N that covers X in H . Therefore, $N \cup M$ induces a path cover in $\mathcal{P}_{1,2}^1(G)$.

However, it is not true that every maximum matching can be extended to a path cover in $\mathcal{P}_{1,2}^1$. For a counterexample for this claim, consider the tree shown in Figure 3.1, where a maximum matching is depicted by the curvy edges.

On the other hand, Proposition 3.2 also says that, if G admits a nontrivial path cover, then it must contain a maximum matching that we can extend. So, in order to find such a matching, we need a structure that encapsulates the information of every maximum matching of a graph. Such

Figure 3.1: A maximum matching that cannot be extended to a path cover in $\mathcal{P}^1_{1,2}$.

structure is given by the well-known Edmonds-Gallai decomposition [LP86] of a graph. Given a graph G , this decomposition is defined by the following partition $[D(G), A(G), C(G)]$ of $V(G)$:

- $D(G)$ is the set of vertices in G which are not covered by at least one maximum matching;
- $A(G)$ is the set of vertices in $V(G) \setminus D(G)$ adjacent to at least one vertex in $D(G)$; and
- $C(G)$ is the set $V(G) \setminus (A(G) \cup D(G))$.

In Figure 3.2 (taken from [LP86]) we show the Edmonds-Gallai decomposition of a graph. Before presenting the theorem that states the properties of this decomposition, we recall some concepts in matching theory. A *near-perfect* matching of a graph G is one covering all but one vertex of G . A graph G is called *hypomatchable* if $G - v$ has a perfect matching for every vertex v in G . The next theorem describes the structure of every maximum matching in G .

Theorem 3.3 (Edmonds-Gallai, 1965). *If G is a graph and $[D(G), A(G), C(G)]$ is the decomposition of G previously defined, then*

- the components of the subgraph induced by $D(G)$ are hypomatchable;*
- the subgraph induced by $C(G)$ has a perfect matching;*
- every maximum matching of G contains a near-perfect matching of each component of $D(G)$, a perfect matching of each component of $C(G)$ and matches all the vertices of $A(G)$ with vertices in different components of $D(G)$.*

We observe that the Edmonds-Gallai decomposition of a graph can be found in polynomial time [LP86]. Now, given a graph G that admits a nontrivial path cover, we define a partition of $V(G)$ in terms of a path cover in $\mathcal{P}^1_{1,2}$. After that, we relate this partition to the Edmonds-Gallai decomposition of G . Let $\mathcal{P}$ be a path cover in $\mathcal{P}^1_{1,2}(G)$. We define a partition of $V(G)$, as follows:

$$\begin{aligned} L(\mathcal{P}) &:= \{v \in V(G) : v \text{ is an end of a 2-path in } \mathcal{P}\}; \\ R(\mathcal{P}) &:= \{v \in V(G) : v \text{ is an internal vertex of a 2-path in } \mathcal{P}\}; \\ S(\mathcal{P}) &:= V(G) \setminus (L(\mathcal{P}) \cup R(\mathcal{P})). \end{aligned}$$

Considering this partition, we prove the following result.

Proposition 3.4. *Let G be a graph that admits a nontrivial path cover, and let $\mathcal{P}$ be a path cover in $\mathcal{P}^1_{1,2}(G)$. Moreover, let $[D(G), A(G), C(G)]$ be the Edmonds-Gallai decomposition of G . Then, the following hold:*

Figure 3.2: a) the Edmonds-Gallai decomposition of a graph G ; b) a maximum matching of G .

- (a) $L(\mathcal{P}) \subseteq D(G)$;
- (b) if $u \in S(\mathcal{P}) \cap A(G)$, then the neighbor of u in $\mathcal{P}$ belongs to $D(G)$;
- (c) let $u, v \in R(\mathcal{P}) \cup S(\mathcal{P})$ (possibly, $u = v$). Let N be the set of neighbors of u and v in $\mathcal{P}$. If $u, v \in A(G)$, then, each vertex in N belongs to a different component in $D(G)$.

Proof. Denote by P_u the path in $\mathcal{P}$ that contains the vertex $u \in V(G)$. First, we show that (a) holds. Let u be a vertex in $L(\mathcal{P})$. Note that, for every edge in P_u we can find a maximum matching that does not contain that edge, therefore $u \in D(G)$. Now, we prove that (b) and (c) hold. Let M be a maximum matching of G obtained by choosing one edge from every path in $\mathcal{P}$.

Let $u \in S(\mathcal{P}) \cap A(G)$. Since $u \in S(\mathcal{P}) \cap A(G)$, the unique edge of P_u belongs to M . Furthermore, by Theorem 3.3, the other end of P_u must belong to $D(G)$. Now, let u and v be vertices that belong to $R(\mathcal{P}) \cup S(\mathcal{P})$. To show (c), we consider three cases depending on whether u or v belong to $S(\mathcal{P})$ or $R(\mathcal{P})$.

Case 1: $u, v \in S(\mathcal{P})$.

In this case, P_u and P_v are 1-paths of $\mathcal{P}$. Let u' and v' be the neighbors of u and v , respectively, in $\mathcal{P}$. Since $uu', vv' \in M$ and $u, v \in A(G)$, by Theorem 3.3, the vertices u' and v' must belong to $D(G)$. If $u = v$, then $u' = v'$ and there is nothing to prove. So, suppose that $u \neq v$. Since M is a maximum matching, then u and v are matched to vertices in different components of $D(G)$. Therefore, u' and v' belong to different components of $D(G)$.

Case 2: $u \in S(\mathcal{P})$ and $v \in R(\mathcal{P})$.

Let u' be the neighbor of u in P_u . Since $v \in R(\mathcal{P})$, it is an internal vertex of a 2-path in $\mathcal{P}$. Let v' and w' be the neighbors of v in P_v , such that $vv' \in M$. We can show that u' and v' belong to different components of $D(G)$ in an analogous way to Case 1. Now, we shall prove that v' and w' belong to different components of $D(G)$. Let K be the component of $D(G)$ that contains v' . Since

the edge vv' belongs to M , by Theorem 3.3, M contains a perfect matching of $K - v'$. This implies that w' must belong to another component of $D(G)$, as w' is not matched in M . Note that an analogous argument shows that u' and w' belong to different components of $D(G)$.

Case 3: $u, v \in R(\mathcal{P})$.

Let u' and t' be the neighbors of u in P_u , such that $uu' \in M$. Also, let v' and w' be the neighbors of v in P_v , such that $vv' \in M$. Using a similar argument as in Case 1, we can show that u' and v' belong to different components of $D(G)$. Also, if we consider a pair of vertices such that one is matched and the other unmatched in M , then we can show that those vertices belong to different components of $D(G)$ using similar arguments as in Case 2. In particular, this shows the case in which $u = v$. Suppose that $u \neq v$. Now, let us prove that t' and w' belong to different components of $D(G)$. Note that, by Theorem 3.3, there is at most one M -unmatched vertex in each component of $D(G)$. Since w' and t' are unmatched, it follows that they must belong to different components of $D(G)$. $\square$

Proposition 3.4 shows how the vertices in $A(G) \cup D(G)$ are covered by a path cover $\mathcal{P}$ in $\mathcal{P}_{1,2}^1(G)$. First, (a) implies that every internal vertex of a 2-path of $\mathcal{P}$ belongs to $A(G) \cup D(G)$. Therefore, the vertices in $C(G)$ are covered by 1-paths induced by a perfect matching of it. Furthermore, items (b) and (c) tell us that a path that contains a vertex in $A(G)$ intersects in a single vertex either one or two components of $D(G)$. Also, the remaining vertices in those components are covered by a set of 1-paths. Finally, for the components in $D(G)$ that are not covered in the previous way, Theorem 3.3 implies that they are covered by a single 2-path and a set of 1-paths. Moreover, we have that these components are nontrivial. Therefore, our existence problem has been reduced to decide, for each path containing a vertex of $A(G)$, which components of $D(G)$ it will intersect. In what follows, we show how to reduce the latter problem to a matching problem in a bipartite graph.

Let $[D(G), A(G), C(G)]$ be the Edmonds-Gallai decomposition of G . Also, let $T \subseteq D(G)$ be the vertices corresponding to trivial components in $D(G)$. Furthermore, consider a set of vertices, say S , that represent the nontrivial components in $D(G) \setminus T$. The **D-reduction** of G is the bipartite graph $G' = (U \cup W, F)$ defined in the following way.

$$\begin{aligned} U &= A(G) \cup A' \cup S', \\ W &= S \cup T, \end{aligned}$$

where A' and S' are copies of the sets $A(G)$ and S , respectively. For every vertex $a \in A(G)$ (resp. $s \in S$), we denote by a' (resp. s') its copy in G' . The set F consists precisely of the following edges: (a) an edge between a vertex $a \in A(G) \cup A'$ and a vertex $w \in W$ if there is an edge, in G , incident to a and to a vertex in the component represented by the vertex w ; (b) an edge linking s to s' , for every $s \in S$. In Figure 3.3, we show an example of a D -reduction.

Now, we state a result from Dulmage & Mendelshon [DM58] that will be useful to prove our next result.

Theorem 3.5 (Dulmage & Mendelshon, 1958). *Let G be an (X, Y) -bipartite graph. If there exist*

Figure 3.3: a) The subgraph induced by $A(G) \cup D(G)$; b) the graph G' .

two matchings, one that covers $A \subseteq X$ and another that covers $B \subseteq Y$, then there exists a matching that covers $A \cup B$.

Finally, we show that the existence of a nontrivial path cover in a graph G is characterized by the existence of a specific matching in the graph G' .

Theorem 3.6. *Let G be a graph and let $G' = (U \cup W, F)$ be the D -reduction of G . Then, G has a nontrivial path cover if, and only if, there is a matching that covers W in G' .*

Proof. First, suppose that G has a nontrivial path cover. Take a path cover $\mathcal{P}$ in $\mathcal{P}_{1,2}^1(G)$. Now, using the structure of $\mathcal{P}$, revealed by Proposition 3.4, we exhibit a matching M in G' that covers W . We note first that, by this proposition, the 2-paths in $\mathcal{P}$ have both ends in $D(G)$. In G' , denote by w_K the vertex in W that represents the component K of $D(G)$. Take w_K in W . Case (1): In $\mathcal{P}$ there is a (unique) 1-path with an end in K and the other end, say a , in $A(G)$. In this case, set the edge $w_K a$ to M . Case (2): In $\mathcal{P}$ there is a 2-path Q that ends in a vertex in K . If Q is entirely contained in K , then set $w_K w'_K$ to M . If the internal vertex of Q , say a , belongs to $A(G)$, then the other end of Q belongs to a component R of $D(G)$ that is distinct from K . In this case, set $w_K a$ and $w_R a'$ to M . Note that, no other 2-path in $\mathcal{P}$ ends in $K \cup R$. Thus, M is a matching in G' that covers W .

Conversely, we show that, if there is a matching in G' that covers W , then there is a nontrivial path cover of G . By Theorem 3.3, every maximum matching in G covers the vertices of $A(G)$ with edges whose other ends belong to different components of $D(G)$. Note that, these edges induce a matching in G' that covers $A(G)$. Then, by Theorem 3.5, there exists a matching that covers the set $A(G) \cup W$ in G' . Let M be such a matching.

Next, we construct a set of edges $F' \subseteq E(G)$ that induces a nontrivial path cover of G . First, let F' be any perfect matching of $C(G)$. Now, let w_K be a vertex of W . We consider two cases.

Case 1: $w_K \in S$ is matched to its copy in S' .

In this case, K is a nontrivial component of $D(G)$. Take a near-perfect matching N of K , and an edge $e \in E(K)$ that is incident to the vertex not covered by N . Add $N \cup \{e\}$ to F' .

Case 2: w_K is matched to a vertex in $A(G) \cup A'$.

Take $u \in A(G) \cup A'$ such that $w_K u \in M$. Let v be a vertex in K that is adjacent to u in G . In this case, consider a perfect matching N of $K - v$ (possibly empty), and add $N \cup \{uv\}$ to F' .

Figure 3.4: a) a nontrivial path cover of $A(G) \cup D(G)$ obtained from b) a matching in G' .

Since M is a matching in G' that covers $A(G) \cup W$, it follows that F' induces a nontrivial path cover of G . $\square$

In Figure 3.4, we show a path cover in $\mathcal{P}_{1,2}^1(G)$ obtained from a matching that covers $A(G) \cup W$ in G' .

Theorem 3.6 shows how we can decide the existence of a nontrivial path cover of a graph. Furthermore, its proof tells us how to obtain a path cover in $\mathcal{P}_{1,2}^1$, if it exists. In what follows, we show a new characterization of the graphs that admit a nontrivial path cover ($[1, 2]$ -factor). Our result resembles the characterization shown by Lovász for (g, f) -factors (see Las Vergnas [LV78]), which we have stated previously as Theorem 3.1.

Most of the characterization results, except for Anstee [Ans85], were not concerned with an efficient way to find a NO-certificate (a set S that does not satisfy the condition stated in Theorem 3.1). Interestingly, our approach of searching for a path cover in $\mathcal{P}_{1,2}^1$ gives an efficient way to find a $[1, 2]$ -factor, when it exists, or to find a NO-certificate. The next theorem tells how the latter can be achieved.

Theorem 3.7. *Let G be a graph, $D(G)$ be the set given by the Edmonds-Gallai decomposition of G , and $T \subseteq D(G)$ be the set of vertices corresponding to the trivial components in $D(G)$. Then the following facts hold:*

- (i) *G has a $[1, 2]$ -factor if and only if $|X| \leq 2|N_G(X)|$, for every $X \subseteq T$.*
- (ii) *If G does not have a $[1, 2]$ -factor, and X is a set that violates the condition stated in (i), then $S = N_G(X)$ is a set that violates the condition stated in Theorem 3.1. Moreover, S can be found in polynomial time.*

Proof. First, we prove (i). Let G' be the D -reduction of G . By Theorem 3.6, it suffices to show that G' has a matching that covers W if and only if $|X| \leq 2|N_G(X)|$, for every subset X of T . First, suppose that G' has a matching that covers W in G' . By Hall's Theorem, for every subset X in W , we have that $|X| \leq |N_{G'}(X)|$. This implies that $|X| \leq |N_{G'}(X)|$ for every set $X \subseteq T$, since $T \subseteq W$. By the definition of G' , we have that $|N_{G'}(X)| = 2|N_G(X)|$, for every $X \subseteq T$. Now, suppose that $|X| \leq 2|N_G(X)| = |N_{G'}(X)|$, for every $X \subseteq T$. By Hall's Theorem, G' has a matching that covers T . Since $N_{G'}(T) \subseteq A(G) \cup A'$, in G' we can match every vertex $s \in S$ to its copy in S' to obtain a matching that covers W .

Now, to show (ii), note that every subset $X \subseteq T$ is an independent set in G . If we consider the set $S = N_G(X)$, then the graph $G - S$ has at least $|X|$ isolated vertices. Thus, if X violates (i), we have that

$$i(G - S) \geq |X| > 2|N_G(X)| = 2|S|.$$

To find a set X that violates (i), we take a maximum matching M' in G' , which clearly does not cover some vertex $t \in T$, and construct in G' an M' -alternating tree rooted at t . The desired set X is composed by the vertices in this tree that belong to T . $\square$

3.3 Optimal nontrivial path covers

Once we verify that a graph has a nontrivial path cover, we may wish to either minimize or maximize the number of paths in the cover. As we mentioned in Chapter 1, we call these problems the MINIMUM NONTRIVIAL PATH COVER (MINNTPC) and the MAXIMUM NONTRIVIAL PATH COVER (MAXNTPC). For ease of reading, we restate their definition.

MINIMUM NONTRIVIAL PATH COVER (MINNTPC): Given a graph G , find a non-trivial path cover $\mathcal{P}$ such that $|\mathcal{P}|$ is minimum, or answer that none exists.

MAXIMUM NONTRIVIAL PATH COVER (MAXNTPC): Given a graph G , find a non-trivial path cover $\mathcal{P}$ such that $|\mathcal{P}|$ is maximum, or answer that none exists.

Since a Hamiltonian path is an optimal solution to MINNTPC in any Hamiltonian graph with at least two vertices, we have that MINNTPC is NP-hard for the class of graphs for which the Hamiltonian path problem is NP-complete.

Now, recall that $\mathcal{P}_{1,2}^1(G)$ consists of path covers of a graph G with the maximum number of 1-paths. Thus, MAXNTPC on G reduces to the problem of finding a path cover in $\mathcal{P}_{1,2}^1(G)$. Next, we show how this can be achieved in polynomial time.

Theorem 3.8. *The MAXNTPC problem can be solved in polynomial time.*

Proof. Let $G = (V, E)$ be a graph. First, we show how to find the Edmonds-Gallai decomposition of G . Note that, we just need to find the set $D(G)$, since $A(G)$ is the set of neighbors of $D(G)$ and $C(G)$ is composed of the remaining vertices of G . Therefore, after finding $D(G)$, we can find $A(G)$ and $C(G)$ in linear time.

Let M be a maximum matching of G , and let $Z \subseteq V$ be the set of M -unmatched vertices. The following characterization of $D(G)$ will be useful for us (see [LP86]):

$$D(G) = \{v \in V : \text{there is an even } M\text{-alternating path from } Z \text{ to } v\}.$$

Thus, after finding a maximum matching M of G , we can find all the vertices that are reachable from Z by an even M -alternating path using a blossom forest with respect to M [KV12]. We can find this forest in linear time. Since we can find a maximum matching of a graph in polynomial time, the same follows for the Edmonds-Gallai decomposition of G .

Suppose that we have already found the decomposition $[D(G), A(G), C(G)]$ and a maximum matching M of G . Note that, we can construct the D -reduction of G in linear time, given $D(G)$ and $A(G)$. Let G' be the D -reduction of G and let M' be a maximum matching of G' . We distinguish two cases.

Case 1: $|M'| = |W|$.

By Theorem 3.6, G admits a nontrivial path cover. Moreover, the proof of this theorem already gives us an algorithm to construct a path cover in $\mathcal{P}_{1,2}^1(G)$ using M' . We observe that the main difficulty in this construction is to find a near-perfect matching that leaves a specific vertex unmatched. We have to find this kind of matching, potentially, for every component of $D(G)$. Now, we show how to find these matchings efficiently using the matching M of G .

Let K be a nontrivial component of $D(G)$. Suppose that we need to find a near-perfect matching of K that leaves a vertex $u \in K$ unmatched. By Theorem 3.3, the matching M induces a near-perfect matching of K in G . Suppose that $v \in K$ is the unique M -unmatched vertex in K . If $u = v$, we have nothing to do. Otherwise, we can use a blossom forest (with respect to M) to find an even M -alternating path between v and u (it exists since K is hypomatchable). Using this path and the matching M , we can obtain the desired matching for the component K . Note that we can do all these computations in $\mathcal{O}(|E|)$ time.

Case 2: $|M'| < |W|$.

In this case G does not admit a nontrivial path cover. The proof of item (ii) in Theorem 3.7 already tells us how to find a negative certificate in linear time using the matching M' . $\square$

The results stated in Theorem 3.6, Theorem 3.7 and Theorem 3.8 guarantee that the algorithm below computes an optimal solution to MAXNTPC.

Algorithm 1 MAXNTPC(G)Input: A graph $G = (V, E)$ Output: A set of edges F that induces a maximum nontrivial path cover of G
or a set of vertices X that violates the condition given in Theorem 3.7

```

1:  $M \leftarrow \text{MAXIMUM-MATCHING}(G)$ 
2:  $[A(G), C(G), D(G)] \leftarrow \text{EDMONDS-GALLAI}(G, M)$ 
3:  $G' \leftarrow D\text{-REDUCTION}(G, D(G), A(G))$ 
4:  $M' \leftarrow \text{MAXIMUM-BIPARTITE-MATCHING}(G')$ 
5: if  $|M'| < |W|$  :
6:    $X \leftarrow \text{FIND-CERTIFICATE}(G, G', M')$ 
7:   return  $(\emptyset, X)$ 
8: else:
9:    $F \leftarrow \text{FIND-MAXNTPC}(G, M, M')$ 
10:  return  $(F, \emptyset)$ 

```

The proof of Theorem 3.8 implies that the running time of Algorithm 1 is dominated by lines 1 and 4. More specifically, by the time required to find a maximum matching in an arbitrary graph and in a bipartite graph. Using the algorithms proposed by Vazirani [Vaz12] and Hopcroft & Karp [HK73], we can solve both problems in $\mathcal{O}(|E| \sqrt{|V|})$ time. Therefore, we obtain the following result.

Corollary 3.9. *Algorithm 1 solves MAXNTPC in $\mathcal{O}(|E| \sqrt{|V|})$ time.*

3.4 Final remarks

We considered path covers that do not contain trivial paths, and studied their existence and optimization. A natural generalization of this problem would be to forbid paths of length less than k , for some positive fixed integer k . Say $Q(k)$ is the problem of finding a path cover in which every path has length at least k . Then $Q(1)$ is the nontrivial path cover problem. It would be interesting to investigate $Q(k)$ for $k \geq 2$.

Regarding $Q(2)$, Wang [Wan94] characterized the bipartite graphs that are feasible instances of this problem. Later, Kano et al. [KKK04] proved an analogous result for general graphs. More recently, Zhang & Zhou [ZZ09] showed an alternative characterization for general graphs, and Zhou et al. [ZWZ17] showed some sufficient conditions for the existence of a path cover by paths of length at least two.

Chapter 4

Minimum path cover as a graph parameter

In this chapter we focus on the MINNTPC problem and its relation with MINPC, especially regarding its complexity and approximability aspects. We begin by presenting in Section 4.1 the results for MINPC that exist in the literature, and then in Section 4.2 we present an algorithm that transforms any path cover into one without trivial paths and without increasing the number of paths in the cover. This result has important implications on the complexity and approximability of MINNTPC. After that, in Section 4.3 we show a linear-time algorithm for MINNTPC on trees, and in Section 4.4, we present some approximability results for MINPC and MINNTPC. We conclude this chapter by presenting two integer programming formulations, one for MINPC and the other for MINNTPC.

4.1 Known results

As we mentioned in Chapter 1, MINPC is NP-hard for many classes of graphs. All these hardness results come from the fact that the Hamiltonian path problem is NP-complete for those classes. The algorithmic results concerning the MINPC problem are mainly concentrated on subclasses of perfect graphs, as shown in Table 4.1. In this table, n and m denote the number of vertices and the number of edges of a graph, respectively.

CLASS OF GRAPH	COMPLEXITY
Trees [FR02]	$\mathcal{O}(n + m)$
Block graph [SSSP93]	$\mathcal{O}(n + m)$
Cactus [MW91]	$\mathcal{O}(n + m)$
Cograph [LOP95]	$\mathcal{O}(n + m)$
Bounded treewidth [ALS91]	$\mathcal{O}(n + m)$
Bipartite permutation [SSSP93]	$\mathcal{O}(n + m)$
Interval [AP90]	$\mathcal{O}(n + m)$
Cocomparability [CDH13]	$\mathcal{O}(\min\{n^2, n + m \log \log n\})$
Distance-hereditary [HC07]	$\mathcal{O}(n^9)$
Circular-arc [HC06]	$\mathcal{O}(n^2 \log n)$
Acyclic digraph [BG77]	$\mathcal{O}(n^{2.5})$

Table 4.1: Polynomial-time algorithms for MINPC in certain classes of perfect graphs.

Moreover, we emphasize that, to the best of our knowledge, there is no approximation algorithm for MINPC. In Figure 4.1, we show the computational complexity status of MINPC in some classes of perfect graphs. For each class in this figure, a shaded rectangle indicates that MINPC is NP-hard, a clear rectangle with solid borderline indicates that $\text{MINPC} \in \text{P}$, and a clear rectangle with dashed borderline indicates that its complexity is unknown. An arrow from a class A to a class B indicates that B is a subclass of class A.

Figure 4.1: Complexity status of MINPC in certain classes of perfect graphs.

Figure 4.2: A graph G with a minimum path cover $\mathcal{P}$ induced by the black edges and the black vertices.

4.2 Cardinality of a nontrivial path cover

In this section, we denote by $\mathcal{G}$ the class of graphs that admit a nontrivial path cover. We denote by $\mu_{nt}(G)$ (resp. $\mu(G)$) the cardinality of a minimum nontrivial path cover (resp. path cover) of a graph G . If $G \notin \mathcal{G}$ then, we say that $\mu_{nt}(G) = +\infty$. Since any nontrivial path cover is a solution to MINPC, we have that $\mu(G) \leq \mu_{nt}(G)$. The results in this section were motivated by the following question:

Is there a graph $G \in \mathcal{G}$ for which $\mu(G) < \mu_{nt}(G)$ or is $\mu(G) = \mu_{nt}(G)$ for every graph $G \in \mathcal{G}$?

We show that $\mu(G) = \mu_{nt}(G)$ for every graph $G \in \mathcal{G}$. The proof of this fact has interesting implications regarding MINPC and MINNTPC.

In what follows we show that, given a graph G and a path cover $\mathcal{P}$ of G , there is a polynomial-time algorithm that either obtains a nontrivial path cover of G , if one exists, or finds a set of vertices of G that violates the condition stated in Theorem 3.1. Moreover, in the former case, the nontrivial path cover obtained by the algorithm has cardinality at most $|\mathcal{P}|$, implying that if $\mathcal{P}$ is a minimum path cover then the nontrivial path cover obtained is also minimum. Before showing this result, we give the intuition behind this algorithm.

First, consider the graph G and the minimum path cover $\mathcal{P}$ of G shown in Figure 4.2. In what follows, we will describe two ways in which we can transform $\mathcal{P}$, by adding and removing edges, to obtain a new path cover with fewer trivial paths. Each one of these transformations corresponds to a path linking a trivial path to a path of length at least three, and whose edges belong alternately to $E(G) \setminus E(\mathcal{P})$ and $E(\mathcal{P})$. We say that a k -path is **long** if $k \geq 3$. In the following figures, the wavy (resp. ticked) edges represent edges that we add to (resp. remove from) $\mathcal{P}$.

The first transformation is depicted in Figure 4.3, where we add an edge linking a trivial path to the long path in $\mathcal{P}$, and remove one of the edges of this long path.

Figure 4.3: a) Modifying $\mathcal{P}$ using an edge linking a trivial path to a long path; and b) the resulting path cover.

Observe that, when a trivial path is adjacent to an internal vertex of a long path (as in the

figure), it does not matter which internal vertex it is adjacent to, we can always add this edge and remove an edge of the long path to obtain a new cover with fewer trivial paths. On the other hand, if a trivial path is not adjacent to a long path, by the minimality of $\mathcal{P}$, it may only be adjacent to an internal vertex of a 2-path. In this case, the only way we can obtain an alternating path is by going through one of the ends of a 2-path, as shown in Figure 4.4. As in the previous case, we can decrease the number of trivial paths using this transformation.

Figure 4.4: a) *Modifying $\mathcal{P}$ using an alternating path from a trivial path to a long path;* and b) *the resulting path cover.*

Observe that, when we do not choose an alternating path, and choose a path that contains exactly one vertex of a 2-path, we may not decrease the number of trivial paths in the cover we obtain. Two examples of this kind of transformation are shown in Figure 4.5.

Figure 4.5: a) *Modifying $\mathcal{P}$;* and b) *the resulting path cover.*

In what follows, we show that these alternating paths are basically the tool we need to transform any path cover into one without trivial paths. We formalize the previous arguments using a directed graph that is defined in terms of $\mathcal{P}$ and G . Now, we give all the necessary definitions and terminology.

If $\mathcal{P}$ denotes a path cover of a graph G , then for every integer $k \geq 0$, we denote by $\mathcal{P}_k$ the set of k -paths in $\mathcal{P}$; and by V_k , the set of vertices spanned by the paths in $\mathcal{P}_k$. In our algorithm, the paths in $\mathcal{P}_2$ play a special role, in particular their internal vertices. The set of such vertices will be denoted by B (as we think of such paths as being of the form $\langle a, b, c \rangle$).

We say that a path cover $\mathcal{P}$ is *good* if no path in $\mathcal{P}$ has an end that is adjacent to an end of another path in $\mathcal{P}$. Clearly, given a path cover that is not good, by joining ends of different paths, we can obtain in polynomial time another path cover (with fewer paths) that is good.

Given a graph G and a good path cover $\mathcal{P}$ of G , we will define a directed graph D , called $\mathcal{P}$ -digraph associated with G , which will help us obtain a nontrivial path cover of G , if one exists.

First, consider the graph $G' = (V', E')$ that results from G by removing the vertices in V_1 and contracting each path in $\bigcup_{k \geq 3} \mathcal{P}_k$ into a single vertex. Let L be the set of vertices that results from

the contraction of those (long) paths. Now, let G^* be the graph obtained from G' by removing all the edges with both ends in L , and also all the edges with an end in B and the other in $B \cup L$.

Figure 4.6: (a) a graph G and $\mathcal{P}$, and (b) the $\mathcal{P}$ -digraph D .

Now we are ready to define the $\mathcal{P}$ -digraph D . It is an orientation of G^* defined in the following way. First, for every vertex $v \in V_0$, orient the edges incident to v as going out from that vertex. Second, for every edge $uv \in E(G^*)$ such that $u \in B$ and $v \in V_2 \setminus B$, orient that edge from v to u if u and v belong to different paths in $\mathcal{P}_2$, otherwise orient it from u to v . Third, orient every edge with one end in $V_2 \setminus B$ and the other in L as going from $V_2 \setminus B$ to L . Finally, orient arbitrarily every edge with both ends in $V_2 \setminus B$. Throughout this section, for each vertex $v \in V$, we denote by P_v the path in $\mathcal{P}$ that contains v . We show in Figure 4.6 the $\mathcal{P}$ -digraph D obtained from a path cover $\mathcal{P}$ represented by a trivial path and the paths consisting of solid edges.

The orientation of the arcs in the $\mathcal{P}$ -digraph D is defined in such a way that the directed paths in D starting at V_0 and ending in a vertex in $B \cup L$, may help finding another path cover of G with fewer trivial paths or confirming the nonexistence of such a path cover.

Lemma 4.1. *Let G be a graph, $\mathcal{P}$ be a good path cover of G , and D be the $\mathcal{P}$ -digraph associated with G . Let $\langle a, b, c \rangle$ be a path in $\mathcal{P}_2$ such that $ac \in E(G)$. If D has a directed path from V_0 to b , then G has a path cover $\mathcal{P}'$ such that $|\mathcal{P}'| < |\mathcal{P}|$ and $|\mathcal{P}'| < |\mathcal{P}_0|$.*

Proof. Let $P = \langle u = r_1, r_2, \dots, r_k = b \rangle$ be a shortest path in D from V_0 to b . We claim that the following hold:

- a) $k = 2m$ for some $m \geq 1$; and
- b) $r_{2i} \in B, r_{2i+1} \in V_2 \setminus B$, for $i = 1, 2, \dots, m - 1$;
- c) r_{2i} and r_{2i+1} belong to the same path in $\mathcal{P}_2$, for $i = 1, \dots, m - 1$.

In fact, considering the construction of D , we have that

- (i) $N^+(v) = \emptyset$, for $v \in L$;

- (ii) $N^+(v) \subseteq B \cup L$, for $v \in V_0$;
- (iii) $N^+(v) \subseteq V_2 \cup L$, for $v \in V_2 \setminus B$;
- (iv) $N^+(v) = V(P_v) \setminus \{v\}$, for $v \in B$.

Since P begins at V_0 and ends at B , by (i), we have that $V(P) \cap L = \emptyset$. Therefore, by (ii), $r_2 \in B$. Moreover, by (iii), (iv), and the minimality of P , if an internal vertex of P belongs to $V_2 \setminus B$, then the next vertex in P belongs to B and vice versa, so the claim follows.

Now, we will construct a path cover of G that has fewer paths than $\mathcal{P}$. Let $Q_i \in \mathcal{P}_2$ be the path that contains r_{2i} , and let s_i be the end of Q_i which is different from r_{2i+1} , for $i = 1, \dots, m-1$. Thus, $Q_m = \langle a, b, c \rangle$. Without loss of generality, suppose that the edge $ac \in E(G)$ is oriented from a to c in D . Consider the paths Q_i^* defined in the following way,

$$Q_i^* := \begin{cases} \langle r_{2i-1}, r_{2i}, s_i \rangle, & \text{for } 1 \leq i \leq m-1, \\ \langle r_{2m-1}, b, a, c \rangle, & \text{for } i = m. \end{cases}$$

Let $\mathcal{P}'$ be the path cover obtained from $\mathcal{P}$ by replacing the paths $Q_1, Q_2, \dots, Q_m, P_u$ with the paths $Q_1^*, Q_2^*, \dots, Q_m^*$. Clearly, $\mathcal{P}'$ is a path cover of G with $|\mathcal{P}'| < |\mathcal{P}|$ and $|\mathcal{P}'_0| < |\mathcal{P}_0|$. $\square$

Theorem 4.2. *Let G be a graph and let $\mathcal{P}$ be a path cover of G containing trivial paths. Then, we can obtain in polynomial time either a nontrivial path cover $\mathcal{P}'$ such that $|\mathcal{P}'| \leq |\mathcal{P}|$, if one exists, or a certificate that G does not have a nontrivial path cover.*

Proof. We assume that $\mathcal{P}$ is a good path cover (as it suffices to prove for this case). Now, consider the $\mathcal{P}$ -digraph D associated with G (that can be constructed in polynomial time). Check whether there is path $\langle a, b, c \rangle$ in $\mathcal{P}_2$ such that $ac \in E(G)$ and D has a directed path from V_0 to b . If this happens, by Lemma 4.1, we can obtain (in polynomial time) a path cover of G with fewer paths and fewer trivial paths. If the new path cover is nontrivial, the proof is complete; otherwise, we repeat this procedure until we obtain a good path cover, say $\widehat{\mathcal{P}}$, that has no 2-path whose ends are adjacent, and whose internal vertex b is the end of a directed path starting at V_0 in the $\widehat{\mathcal{P}}$ -digraph associated with G . We say that such a path cover is *very good*. Clearly, it can be obtained in polynomial time. For ease of notation, let us call $\mathcal{P}$ the very good path cover obtained this way, and consider the $\mathcal{P}$ -digraph D associated with G . If $\mathcal{P}$ contains trivial paths, then one of the following two cases must occur.

Case 1: there exists a directed path from V_0 to L in D .

Let P be such a path with minimum length. By arguments similar to those used in the proof of Lemma 4.1, we can consider that

$$P := \langle u, x_1, y_1, \dots, x_m, y_m, v \rangle,$$

where $u \in V_0$, $v \in L$, $x_i \in B$, $y_i \in V_2 \setminus B$ and $m \geq 0$. Without loss of generality, consider that $m \geq 1$ (the case in which $m = 0$ can be dealt in a similar way). We recall that P_v denotes the path in $\mathcal{P}$

that was contracted to v . Let w be a vertex of P_v that is a neighbor of y_m in G . Since y_m is an end of a path in $\mathcal{P}$, and $\mathcal{P}$ is good, we conclude that w is an internal vertex of P_v . Let P'_v be a shortest subpath of P_v containing w and an end of P_v . Also, consider that $Q'_v = P_v \setminus V(P'_v)$. Since $|P_v| \geq 3$, we have that $|Q'_v| \geq 1$.

Let $Q_i \in \mathcal{P}_2$ be the path that contains x_i , and let z_i be the end of Q_i which is different from y_i , for $i = 1, 2, \dots, m$. Now, we define the paths Q_i^* in the following way,

$$Q_i^* = \begin{cases} \langle u, x_1, z_1 \rangle, & \text{if } i = 1, \\ \langle y_{i-1}, x_i, z_i \rangle, & \text{if } 1 < i \leq m, \\ \langle y_m \rangle \cdot P'_v, & \text{if } i = m + 1. \end{cases}$$

Let $\mathcal{Q}$ be the path cover of G obtained from $\mathcal{P}$ when we replace the paths in $\{P_u, Q_1, \dots, Q_m, P_v\}$ by the paths in $\{Q_1^*, \dots, Q_{m+1}^*, Q'_v\}$. Observe that $|\mathcal{Q}| = |\mathcal{P}|$ and $|\mathcal{Q}_0| = |\mathcal{P}_0| - 1$. In Figure 4.7, we show an example considering the path cover $\mathcal{P}$ and the $\mathcal{P}$ -digraph D shown in Figure 4.6.

Figure 4.7: (a) a path P ; and (b) the resulting path cover $\mathcal{Q}$.

Case 2: there is no directed path from V_0 to L in D .

Let u be a vertex in V_0 . If $N_G(u) = \emptyset$, then $S = \emptyset$ violates the condition stated in Theorem 3.1. So, suppose that $N_G(u) \neq \emptyset$. Since there is no directed path from V_0 to L , and $\mathcal{P}$ is good, we have that $N_D^+(u) \subseteq B$. Let B_u be the set of vertices of B that are reachable by a directed path from u .

Let $S := B_u$. In what follows, we show that S violates the condition stated in Theorem 3.1. First, note that u is an isolated vertex in $G - S$. Now, let T_u be the set of the ends of the paths in $\mathcal{P}_2$ whose internal vertex belongs to B_u . We will show that T_u is an independent set in $G - S$. Let t be any vertex in T_u . Since there is no directed path from u to L in D , we have that t is not adjacent, in G , to any vertex in V_k , $k \geq 3$. Since $\mathcal{P}$ is very good, we have that t is not adjacent, in G , to any vertex in $V_2 \setminus S$. (Note that, if t is adjacent to an internal vertex b of another path in $\mathcal{P}_2$, then b is also reachable by a directed path from u and, thus, b belongs to S .)

Moreover, since $\mathcal{P}$ is good, vertex t is not adjacent to any vertex in $V_0 \cup V_1$. Therefore, t is an isolated vertex in $G - S$. Since $G - S$ has at least $2|S| + 1$ isolated vertices, by Theorem 3.1, the

graph G does not have a nontrivial path cover.

If Case 1 occurred, and the new path cover still contains trivial paths, we repeat the procedure. As, each time this procedure is repeated we obtain a path cover with fewer trivial paths, we either find a nontrivial path cover of G , or Case 2 occurs and we find a nonexistence certificate. All these steps can be done in polynomial time. $\square$

By Theorem 4.2, the following results follow.

Corollary 4.3. *If a graph G admits a nontrivial path cover, then $\mu_{nt}(G) = \mu(G)$.*

Corollary 4.4. *The problems MINPC and MINNTPC have the same computational complexity in $\mathcal{G}$ (the class of graphs that admit a nontrivial path cover).*

4.3 Minimum nontrivial path cover on trees

As we mentioned, MINPC on trees can be solved in linear time. Thus, given a tree G , we can take a minimum path cover of G and use Theorem 4.2 to either transform it into a minimum nontrivial path cover of G , if it exists; or exhibit a NO-certificate. Instead of using such a procedure, we show a simpler linear-time algorithm for MINNTPC on trees, which can be easily extended for the edge-weighted version of the problem (presented in the next chapter).

To understand the idea behind the algorithm, given a tree T , suppose that T has a minimum nontrivial path cover $\mathcal{P}$. Consider that T' is the arborescence obtained when we root T at a vertex r in T . Let T'_u be the subarborescence of T' rooted at u . Also, let T_u be the subtree of T induced by the vertices in T'_u . Note that, when we consider the subgraph of T spanned by the edges of $\mathcal{P}$, the vertex u has degree 0, 1 or 2 in T_u .

Let $f(u, d)$ be the cardinality of a minimum nontrivial path cover of T_u , in which the vertex u has degree d . Algorithm 2, described below, computes the values $f(u, d)$ in post-order using a DFS traversal of T (defined later). This algorithm is implemented by a recursive procedure ($\text{DFS}(T, v, \text{parent})$) that receives as first parameter the tree T , as second parameter the current vertex in the transversal, and as third parameter, its parent in the arborescence. Thus, to find the optimal value of MINNTPC in T , we choose r to be any leaf of T , and call $\text{DFS}(r, \text{nil})$. At the end, if T has a nontrivial path cover, then $f(r, 1) < +\infty$, and it indicates the cardinality of a minimum nontrivial path cover. We may retrieve the edges of such a cover by modifying slightly this algorithm.

Algorithm 2 DFS(T, v, parent)

Input: A tree T , the vertex v that is being explored, and its parent in T

Output: Values $f(v, d)$ for every vertex v in T and integer d in $\{0, 1, 2\}$

```

1:  $\text{deg} \leftarrow 0$ 
2:  $m_1 \leftarrow +\infty$ 
3:  $m_2 \leftarrow +\infty$ 
4:  $\text{sum} \leftarrow 0$ 
5: for  $u \in N(v)$  :
6:   if  $u \neq \text{parent}$  :
7:     DFS( $T, u, v$ )
8:      $Y \leftarrow \min(f(u, 1), f(u, 2))$ 
9:      $X \leftarrow \min(f(u, 1), f(u, 0))$ 
10:     $\text{sum} \leftarrow \text{sum} + Y$ 
11:     $\text{deg} \leftarrow \text{deg} + 1$ 
12:    if  $m_1 > X - Y$  :
13:       $m_2 \leftarrow m_1$ 
14:       $m_1 \leftarrow X - Y$ 
15:    else if  $m_2 > X - Y$  :
16:       $m_2 \leftarrow X - Y$ 
17: for  $d = 0$  to  $2$  :
18:    $f(v, d) \leftarrow +\infty$ 
19: if  $\text{deg} \geq 0$  :
20:    $f(v, 0) \leftarrow \text{sum} + 1$ 
21: if  $\text{deg} \geq 1$  :
22:    $f(v, 1) \leftarrow \text{sum} + m_1$ 
23: if  $\text{deg} \geq 2$  :
24:    $f(v, 2) \leftarrow \text{sum} + m_1 + m_2 - 1$ 

```

Theorem 4.5. Algorithm 2 correctly computes $f(v, d)$ for every $v \in V(T)$ and $d \in \{0, 1, 2\}$.

Proof. Let v be any vertex in T . We prove the statement by induction on $n := |V(T_v)|$. If $n = 1$, then v is a leaf of the tree. In this case, at the end of the loop starting at line 5, we have that $\text{sum} = \text{deg} = 0$. Therefore, at the end of the algorithm $f(v, 0) = 1$ and $f(v, 1) = f(v, 2) = +\infty$.

Now, suppose that $n \geq 2$. Let $v_1, v_2, \dots, v_k$ be the neighbors of v in T_v . First, if $d > k$, then there is no solution to $f(v, d)$. Therefore, in line 17, we set $f(v, d) = +\infty$, and this value is not changed afterwards. So, let us suppose that $d \leq k$. Since we are restricting the path covers to those in which v has degree d in T_v , we have to choose d edges incident to v and v_i , $1 \leq i \leq k$, to belong to the path cover. Observe that for the vertices v_i such that vv_i belongs to the path cover, the degree in its corresponding subtree T_{v_i} can be zero or one. In case vv_i does not belong to the cover, the degree of v_i in T_{v_i} must be one or two. For $i = 1, 2, \dots, k$, let $X_{v_i} := \min\{f(v_i, 1), f(v_i, 0)\}$ and

let $Y_{v_i} := \min\{f(v_i, 1), f(v_i, 2)\}$. Now, we show how to express $f(v, d)$ in terms of X_{v_i} and Y_{v_i} .

First, let us suppose that $d = 2$, and let v_a and v_b be neighbors of v in a minimum path cover where v has degree two in T_v . By the previous arguments, we have that

$$f(v, 2) = \sum_{\substack{i=1 \\ i \neq a, b}}^k Y_{v_i} + X_{v_a} + X_{v_b} - 1.$$

By adding and subtracting Y_{v_a} and Y_{v_b} , we get the following expression,

$$f(v, 2) = \sum_{i=1}^k Y_{v_i} + (X_{v_a} - Y_{v_a}) + (X_{v_b} - Y_{v_b}) - 1.$$

Since $\sum_{i=1}^k Y_{v_i}$ is a constant, to compute the value of $f(v, 2)$, we need to find two neighbors of v that minimize the sum of the corresponding $X_{v_i} - Y_{v_i}$. Observe that at the end of the loop starting at line 5, the variable *sum* is equal to $\sum_{i=1}^k Y_{v_i}$, and m_1 and m_2 hold the desired minima. By induction hypothesis our algorithm correctly computes the values of $f(v_i, d)$ for $1 \leq i \leq k$, $1 \leq d \leq 2$. Therefore, the algorithm correctly computes the value $f(v, 2)$. The cases in which $d \in \{0, 1\}$ can be shown by analogous arguments. $\square$

Since we process each vertex v of the tree just once, and we iterate through its neighbors to compute the values of $f(v, d)$, the complexity of our algorithm is $\mathcal{O}(n)$, where n is the number of vertices of the tree. Thus, we obtain the following result.

Corollary 4.6. *MINNTPC on trees can be solved in linear time.*

4.4 Approximability of MINPC and MINNTPC

In this section we present some approximability results for MINPC and MINNTPC. Note that, since the cardinality of any path cover is an integer number, for any $\epsilon > 0$, there cannot exist a $(2-\epsilon)$ -approximation for MINPC or MINNTPC, otherwise such algorithm would solve the Hamiltonian path problem in polynomial time. Thus, we obtain the following result

Corollary 4.7. *For every $\epsilon > 0$, there is no $(2 - \epsilon)$ -approximation for MINPC and MINNTPC, unless $P \neq NP$.*

Now, consider a special variant of the MINIMUM TRAVELING SALESMAN problem, in which the edge-weights are 1 or 2, denoted as MINTSP-(1,2). Papadimitriou and Yannakakis [PY93] showed that this problem is Max SNP-hard.

Vishwanathan [Vis92] showed a relation between the approximation ratios of MAXPC (to be stated in Chapter 5) and MINTSP-(1,2). In a similar way, we show an analogous relation between MINPC and MINTSP-(1,2).

Proposition 4.8. *If there is an α -approximation algorithm for MINPC, then there is a $(1 + \alpha)/2$ -approximation algorithm for MINTSP-(1,2).*

Proof. Let (K, w) be a complete graph on n vertices whose edge-weights, given by w , are 1 or 2. Let G be the spanning subgraph of K that contains only the edges of weight one. Let $\mathcal{A}$ be an α -approximation for MINPC, $\alpha \geq 1$, and let $\mathcal{P}$ be a path cover of G returned by Algorithm $\mathcal{A}$. Thus,

$$|\mathcal{P}| \leq \alpha |\mathcal{O}|, \quad (4.1)$$

where $\mathcal{O}$ is a minimum path cover of G . Linking the paths of $\mathcal{P}$ (resp. $\mathcal{O}$) we can obtain a cycle in K of weight at most $n + |\mathcal{P}|$ (resp. $n + |\mathcal{O}|$). From (4.1), we have that $n + |\mathcal{P}| \leq n + \alpha |\mathcal{O}|$. Finally, the approximation ratio of such algorithm is given by the following expression.

$$\inf \left\{ r \geq 1 : \frac{n + \alpha |\mathcal{O}|}{n + |\mathcal{O}|} \leq r, 1 \leq |\mathcal{O}| \leq n \right\},$$

whose optimal value is attained when $r = (1 + \alpha)/2$. $\square$

Now, using the results obtained in the previous section, we show the equivalence between MINPC and MINNTPC in terms of approximability.

Corollary 4.9. *Let $\mathcal{G}$ be the class of graphs that admit a nontrivial path cover. There is an α -approximation for MINPC in $\mathcal{G}$ if, and only if, there is an α -approximation for MINNTPC in $\mathcal{G}$.*

Proof. First, observe that if there is an α -approximation for MINNTPC, then by Corollary 4.3, this is also an α -approximation for MINPC. Now, let $\mathcal{A}$ be an α -approximation for MINPC, and let $\mathcal{P}$ be a path cover returned by $\mathcal{A}$ when applied to a graph $G \in \mathcal{G}$. By Theorem 4.2, we can obtain in polynomial time a nontrivial path cover $\mathcal{P}'$ of G such that $|\mathcal{P}'| \leq |\mathcal{P}|$. Let $\mathcal{O}$ be a minimum nontrivial path cover of G . By Corollary 4.3, we have that $|\mathcal{P}| \leq \alpha |\mathcal{O}|$. Therefore, $|\mathcal{P}'| \leq \alpha |\mathcal{O}|$, and hence there is also an α -approximation for MINNTPC in $\mathcal{G}$. $\square$

4.5 Integer formulations for MINPC and MINNTPC

In this section we present two integer formulations, one for MINPC and the other for MINNTPC. To the best of our knowledge, these are the first integer formulations for these problems.

4.5.1 Description of the formulations

Let $G = (V, E)$ be a graph. Consider the graph $G' = (V', E')$ obtained from G in the following way

$$\begin{aligned} V' &= V \cup \{s, t\}, \\ E' &= E \cup \{sv, vt : v \in V\}. \end{aligned}$$

In other words, G' is obtained from G by adding two vertices, s and t , and by linking those vertices to every vertex of G . Now, let $\mathcal{P}$ be any path cover of G . Observe that, if for each path of $\mathcal{P}$, we link one end to s and the other to t , we obtain a set of internally-disjoint paths from s to t that cover V . In an analogous way, if we have a set of internally-disjoint paths from s to t that contain

every vertex in V , after removing s and t , we obtain a path cover of G . Therefore, we want to obtain a subgraph, say H , of G' such that

- a) H is connected,
- b) $|\delta_H(v)| = 2$, for each $v \in V$,
- c) $|\delta_H(s)| = |\delta_H(t)|$, and
- d) $|\delta_H(s)|$ is as small as possible.

So, we model MINPC by the following integer linear program, defined on $G' = (V', E')$. The binary variable $x = (x_e)_{e \in E'}$ in the formulation is used to indicate which edges of G' are selected (to belong to H). More precisely, $x_e = 1$ if and only if e is selected.

$$\begin{aligned}
 & \text{minimize} && x(\delta(s)) \\
 & \text{subject to} && x(\delta(v)) = 2, && \forall v \in V, \\
 & && x(\delta(s)) = x(\delta(t)), \\
 & && x(\delta(S)) \geq 1 && \forall S \subset V', S \neq \emptyset \\
 & && x_e \in \{0, 1\}, && \forall e \in E'.
 \end{aligned} \tag{4.2}$$

In case of MINNTPC, since paths of the form $\langle s, v, t \rangle$ are precisely trivial paths in G , we just need to impose that $x(sv) + x(vt) \leq 1$ for every $v \in V$. Thus, we obtain the following ILP.

$$\begin{aligned}
 & \text{minimize} && x(\delta(s)) \\
 & \text{subject to} && x(\delta(v)) = 2, && \forall v \in V, \\
 & && x(sv) + x(vt) \leq 1, && \forall v \in V, \\
 & && x(\delta(s)) = x(\delta(t)), \\
 & && x(\delta(S)) \geq 1 && \forall S \subset V', S \neq \emptyset \\
 & && x_e \in \{0, 1\}, && \forall e \in E'.
 \end{aligned} \tag{4.3}$$

Let H be the subgraph induced by the support of a feasible solution x of (4.2). Note that, the first two inequalities in (4.2) ensure that $|\delta_H(v)| = 2$, for every $v \in V$, and $|\delta_H(s)| = |\delta_H(t)|$. But, these two inequalities do not guarantee that H is connected. For example, if G is a Hamiltonian graph, and we omit the inequality

$$x(\delta(S)) \geq 1 \quad \forall S \subset V', S \neq \emptyset, \tag{4.4}$$

in (4.2), an optimal solution would induce a graph consisting of three components: two trivial components (the vertices s and t) and the third one being exactly a Hamiltonian cycle of G . Observe that inequality (4.4) ensures that a feasible solution x induces a connected graph H . Otherwise, let $S \subset V'$ be the set of vertices spanned by a component of H . Since $\delta(S) = \emptyset$, x violates (4.4). Inequality (4.4) is known as the *cutset constraint*.

Since we have an exponential number of inequalities in both formulations, we need a separation routine for inequality (4.4). Given a vector x , to verify whether x satisfies inequality (4.4), we can find a minimum cut of the subgraph of G' induced by the support of x . Let G^* be such a subgraph. Although finding a minimum cut of a graph can be solved by a maximum flow algorithm, this can be computationally expensive when compared to graph search algorithms. In our implementation of (4.2) and (4.3), we take advantage of the integrality constraint and implement the separation routine using a depth first search (DFS) that finds every component of G^* . In case G^* is not connected, for each component, we add an inequality as in (4.4) with S corresponding to the vertices spanned by that component.

4.5.2 Computational environment

We implemented the integer formulations in C++ using the Gurobi optimization software [Gur19], version 7.5.1 for 64 bits architectures. In Table 4.2, we give the specifications of the computer used to run the experiments.

Processor	
Model	Intel Core i5
Clock speed	2.3 GHz
Number of cores	2
Cache L2	256 KB (per core)
Cache L3	3 MB
RAM memory	
Size	2 GB (4 GB in total)
Type	DDR3
Clock speed	1333 MHz
Other	
O.S.	Mac OS X 10.11.6 (El Capitan)

Table 4.2: *Computational environment.*

4.5.3 Description of the experiments

We tested both formulations on two sets of instances. The first set of instances is composed of random graphs constructed using the Erdős & Rényi model $\mathcal{G}(n, p)$ [ER59], where n is the number of vertices and p is the probability that an edge exists in the graph. The second set of instances is composed of graphs constructed using the Barabási & Albert model [BA99]: they have a power-law (or scale-free) degree distribution. All the instances were generated using the NetworkX package [HSS08] for Python. Next, we describe how we chose the parameters to generate these instances.

The random graphs that are generated using the Erdős & Rényi model have interesting properties. In particular, the following result is of our interest in this context (see Bollobás [Bol98]).

Theorem 4.10 (Kömlos & Szemerédi, 1983). *Consider the space $\mathcal{G}(n, p)$ and let $p^*(n) = \frac{\ln n + \ln \ln n}{n}$. Then $p^*(n)$ is the threshold function for the property of being Hamiltonian.*

Because of Theorem 4.10, we decided to study the behavior of our formulations in graphs whose density is around this threshold. Our intention was to investigate whether the property of being Hamiltonian has an impact on the running time of our implementations. We considered graphs of order $n = 100, 300, 500$, and 700 in these experiments. In Table 4.3, we show the value of the threshold function and the five densities we considered for each value of n .

n	$p^*(n)$	Density considered
100	0.0613	4%, 5%, 6%, 7%, 8%
300	0.0248	1%, 2%, 3%, 4%, 5%
500	0.0161	1%, 2%, 3%, 4%, 5%
700	0.0120	1%, 2%, 3%, 4%, 5%

Table 4.3: *Hamiltonicity threshold $p^*(n)$ for some values of n .*

On the other hand, we also tested both formulations in denser graphs. For this, we vary each parameter in the following way

- Order of the graph (n): 100, 300, 500, 700, and 900.
- Density (p): 30%, 50%, and 70%.

For each combination of n and p , we generated 20 instances and computed the average running time.

As we mentioned in the beginning of this section, the second set of instances follows the Barabási & Albert model [BA99]. Every graph in this model depends on two parameters:

- n : the order of the graph, and
- m : the number of edges that link a vertex that is added to the graph.

To create a graph in this model, we begin with a graph with m vertices and no edges. While the number of vertices is less than n , we add a vertex and link it to m distinct vertices. Let u be the vertex that is being added to the graph, and let v be any vertex in the current graph. Then, the probability that we link u and v is proportional to the degree of v in the current graph. This process guarantees that, at the end, we obtain a graph that has a power-law degree distribution (this does not happen in the Erdős & Rényi model).

In our experiments, we considered graphs of order $n = 300, 500$, and 700 . In case of the parameter m , we partition the set of possible values $\{1, 2, \dots, n-1\}$ into three roughly equal-size sets. That is, we considered three intervals $[\ell_1, r_1]$, $[\ell_2, r_2]$ and $[\ell_3, r_3]$ such that

$$r_i = \begin{cases} \lceil \frac{i}{3}n \rceil, & \text{if } i = 1, 2; \\ n-1, & \text{otherwise;} \end{cases}$$

and

$$\ell_i = \begin{cases} 1, & \text{if } i = 1; \\ r_{i-1} + 1, & \text{otherwise.} \end{cases}$$

Finally, for each combination of n and interval $[\ell_i, r_i]$, we generated 20 instances choosing a value of m inside that interval uniformly at random.

The reader interested in the implementation, instances, and results, can consult the public repository

<https://github.com/renzov/path-cover>

4.5.4 Experimental results

In this section, we present the results of our experiments. We measured the running time of our implementations in seconds. Throughout this section, we will refer to the formulation (4.2) (resp. (4.3)) as TP (resp. NTP). First, we consider the instances that follow the Erdős & Rényi model.

The results for graphs whose density is close to the Hamiltonicity threshold (given by Theorem 4.10) are shown in Table 4.4 and Table 4.5. In each case we present two tables, one containing the information regarding the running time, and the other containing the information regarding the optimal value that was found. Moreover, the five densities we considered are represented by p_i , for $i = 1, \dots, 5$, where p_1 (resp. p_5) represents the lowest (resp. highest) density as shown in Table 4.3.

n	p_1	p_2	p_3	p_4	p_5
100	0.016	0.032	0.061	0.099	0.121
300	0.021	0.195	0.595	0.956	1.093
500	0.110	3.486	2.308	5.032	5.400
700	1.607	6.861	9.752	9.859	13.943

(a) *Running time.*

n	p_1	p_2	p_3	p_4	p_5
100	6.30	2.35	1.60	1.10	1.10
300	45.20	3.05	1.05	1.00	1.00
500	12.55	1.00	1.00	1.00	1.00
700	3.00	1.00	1.00	1.00	1.00

(b) *Objective value.*

Table 4.4: *Results obtained by TP.*

In Figure 4.8, we show a line chart that represents the information given in Table 4.4a. As we can see in this figure, there is a sudden increase in the running time when we surpass the Hamiltonicity threshold. This is more notorious in the case of $n = 500$ and $n = 700$, when the density increases from 1% to 2%. This would suggest that instances that contain a Hamiltonian cycle (path) are harder to solve for TP. Next, we present the results corresponding to NTP.

In the case of NTP, our implementation found infeasible instances usually when the density was below the Hamiltonicity threshold. Note that, to calculate the (average) objective value, we considered only feasible instances. Moreover, we write a dash (–), when no feasible instances were found. We show the information regarding the number of feasible instances in Table 4.6.

We show in Figure 4.9 a line chart corresponding to the running time of NTP given in Table 4.5a. Mostly, we cannot see a notorious increase in the running time of NTP when the density surpasses

Figure 4.8: Running time of TP around the Hamiltonicity threshold.

n	p_1	p_2	p_3	p_4	p_5
100	0.005	0.028	0.051	0.061	0.116
300	0.004	0.076	0.931	2.006	1.075
500	0.005	2.514	2.667	3.902	5.771
700	0.886	11.366	8.494	18.086	15.162

(a) Running time.

n	p_1	p_2	p_3	p_4	p_5
100	4.40	1.92	1.08	1.00	1.00
300	—	2.27	0.95	1.00	1.00
500	—	1.00	1.00	1.00	1.00
700	1.75	1.00	1.00	1.00	1.00

(b) Objective value.

Table 4.5: Results obtained by NTP.

the threshold. The only case this happens is when $n = 700$ and the density increases from 1% to 2%. This may be caused by the variation on the number of infeasible instances between data sets, since this also affects the average running time.

In what follows, we present the results obtained for TP and NTP when executed on graphs with higher density. In virtue of Theorem 4.10, all these graphs contain a Hamiltonian cycle with high probability. Indeed, after checking the solutions found by our implementations, we found that every graph in these data sets contains a Hamiltonian path.

We show separately the results obtained for the different densities. For each value of p , we show a table containing the information of all the experiments for that value. Moreover, each table contains the following columns.

Figure 4.9: Running time of NTP around the Hamiltonicity threshold.

n	p_1	p_2	p_3	p_4	p_5
100	5	12	13	18	18
300	0	11	19	20	20
500	0	20	20	20	20
700	8	20	20	20	20

Table 4.6: Number of instances containing a nontrivial path cover in the Erdős & Rényi model.

- n : the number of vertices of the graph
- Time: the average running time (in seconds)
- SD: the standard deviation of the running time
- Min: the lowest running time attained
- Max: the highest running time attained

We show in Table 4.7, Table 4.8, and Table 4.9 the results obtained for densities 30%, 50%, and 70%, respectively.

n	Average	Min.	Max.	SD
300	9.58	0.47	43.39	11.49
500	22.25	1.89	50.88	13.76
700	52.88	5.40	293.70	65.01
900	121.02	18.07	492.14	112.85

(a) TP

n	Average	Min.	Max.	SD
300	9.12	0.72	66.60	14.53
500	31.97	1.87	126.23	27.60
700	57.83	6.03	265.23	78.76
900	149.77	4.87	1020.76	221.20

(b) NTP

Table 4.7: Results obtained for $p = 30\%$.

n	Average	Min.	Max.	SD
300	7.82	0.30	36.08	10.34
500	28.02	1.60	115.15	32.99
700	166.12	13.95	771.82	174.96
900	321.63	38.87	891.30	271.49

(a) TP

n	Average	Min.	Max.	SD
300	9.08	0.50	24.32	7.99
500	33.79	3.38	164.22	40.06
700	213.32	8.29	2273.17	499.38
900	266.01	21.17	977.07	287.67

(b) NTP

Table 4.8: Results obtained for $p = 50\%$.

We show in Figure 4.10 a line chart corresponding to the results obtained for TP. Observe that the running time exhibits an exponential growth. Furthermore, as n grows, the instances with $p = 50\%$ seem harder to solve on average.

The information on the running time of NTP on dense graphs is depicted in Figure 4.11. As in the case of TP, we see that the graphs with density $p = 50\%$ are harder to solve for this formulation. Also, the cases $p = 30\%$ and $p = 70\%$ present similar running times, the latter being a little higher.

Before presenting the second set of experiments, observe that there is a considerable difference between the minimum and maximum running time attained in each data set. This happens in

Figure 4.10: *Running time of TP on dense graphs.*

Figure 4.11: *Running time of NTP on dense graphs.*

n	Average	Min.	Max.	SD
300	8.70	0.45	26.65	7.68
500	43.08	5.74	198.23	44.39
700	165.85	9.39	598.79	190.83
900	195.56	12.86	952.87	244.02

(a) TP

n	Average	Min.	Max.	SD
300	10.45	0.61	31.04	8.45
500	50.96	4.19	214.05	52.90
700	104.61	6.31	389.08	112.07
900	159.29	12.26	533.62	134.64

(b) NTP

Table 4.9: Results obtained for $p = 70\%$.

both formulations, and is exhibited by the standard deviation (SD) in each table. Although, both formulations have a good performance on average, there is much to investigate about the instances that are harder to solve for them.

In what follows, we show the results obtained by TP and NTP on the graphs that obey the Barabási & Albert model. The instances generated by this model for the same parameter m have the same number of edges. For each formulation we present two tables, one containing the information regarding the running time, and the other containing the information regarding the optimal value that was found. Furthermore, in those tables, the column m_i represents the interval $[\ell_i, r_i]$ as described in the previous section. Table 4.10 (resp. Table 4.11) contains the results obtained for TP (resp. NTP).

n	m_1	m_2	m_3
300	2.44	5.99	0.07
500	22.48	16.93	0.29
700	62.82	26.53	0.85

(a) Running time.

n	m_1	m_2	m_3
300	1.00	19.05	232.70
500	1.00	46.25	363.90
700	19.40	40.25	456.30

(b) Objective value.

Table 4.10: Results obtained for TP.

We show in Figure 4.12 a line chart that compares the running time of TP for the different values of m . In this chart, we can see that the running time of TP decreases as m grows. On the other hand, the objective value (number of paths) increases with the growth of m . This would suggest that graphs that need many paths to be covered are solved more easily by our formulation. Note that, this is also the case for graphs of low density in the Erdős & Rényi model (as shown by Table 4.4a). Next, we comment on the results obtained by NTP.

n	m_1	m_2	m_3
300	3.64	9.09	0.09
500	24.56	44.89	0.30
700	52.05	59.09	0.91

(a) Running time.

n	m_1	m_2	m_3
300	1.00	19.05	—
500	1.00	46.25	—
700	1.00	40.25	—

(b) Objective value.

Table 4.11: Results obtained for NTP.

In the case of NTP, our implementation found infeasible instances, especially when the value of the parameter m was high. Note that, to calculate the (average) objective value, we considered

Figure 4.12: Running time for TP in graphs that obey the Barabási & Albert model.

Figure 4.13: Running time for NTP in graphs that obey the Barabási & Albert model.

only feasible instances. Moreover, we write a dash (–), when no feasible instances were found. We show the information regarding the number of feasible instances in Table 4.12.

In Figure 4.13, we show a line chart comparing the running time of NTP for different values of m . This figure suggests that instances in which m is around $n/2$ are harder to solve on average. On the other hand, the running time of NTP was low on infeasible instances, which was also the case for graphs in the Erdős & Rényi model (see Table 4.5a).

n	m_1	m_2	m_3
300	20	20	0
500	20	20	0
700	19	20	0

Table 4.12: Number of instances containing a nontrivial path cover in the Barabási & Albert model.

To conclude this section, we comment on the results obtained by our experiments. It seems that our formulations perform better on graphs that do not admit a nontrivial path cover or need many paths to be covered. Furthermore, the graphs that contain a Hamiltonian path seem to be harder to be solved. Finally, we observed that on those (Hamiltonian) graphs, the running times were not so

similar: an indication that, besides Hamiltonicity, other properties of the graph have great influence on the running time of our formulations.

4.6 Final remarks

It is still an open problem to find an approximation algorithm for MINPC (MINNTPC). Note that, the optimal value of the linear relaxation of the formulation (4.2) gives us a lower bound for the optimal value of MINPC. Using Gurobi, we implemented this relaxation. Unfortunately, we could not infer any useful information about the structure of the subgraphs induced by the support of fractional vertices of this relaxation. A further study of this relaxation may lead to the design of an approximation algorithm for MINPC.

Chapter 5

Weighted path covers

In this chapter we focus on the weighted versions of the path cover problem. In these problems, a graph with nonnegative weights on the edges is given, and we seek a path cover with maximum or minimum total weight. When trivial paths are allowed, the minimization problem is easy, since a path cover consisting only of trivial paths is an optimal solution. For the maximization problem, called **MAXIMUM PATH COVER (MAXPC)**, we present in Section 5.1 some known results from the literature, including some constant factor approximation algorithms. After that, in Section 5.2 we consider weighted nontrivial path covers. We call the corresponding problems the **MINIMUM WEIGHT NONTRIVIAL PATH COVER (MINWNTPC)** and the **MAXIMUM WEIGHT NONTRIVIAL PATH COVER (MAXWNTPC)**. We show that MINWNTPC can be solved in polynomial time, while MAXWNTPC is NP-hard. We also show a $(2/3)$ -approximation algorithm for MAXWNTPC. Finally, in Section 5.4 we study the polytope defined by the incidence vectors of the $[1, 2]$ -factors of a graph, and show some valid inequalities for it.

5.1 Known results

In the case of the weighted version of the path cover problem, research has focused on finding path covers of maximum total weight. We define formally this problem as follows:

MAXIMUM PATH COVER (MAXPC): Given a graph $G = (V, E)$ with nonnegative weight w_e for each edge e , find a path cover $\mathcal{P}$ such that $w(\mathcal{P})$ is maximum.

First, we note that MINPC is equivalent to MAXPC on graphs with uniform (or unit) weights. To show this, consider a graph $G = (V, E)$ with weight $w_e = 1$, for all $e \in E$. If $\mathcal{P}$ is a path cover of G , then

$$w(\mathcal{P}) = \sum_{P \in \mathcal{P}} w(E(P)) = \sum_{P \in \mathcal{P}} (|V(P)| - 1) = |V| - |\mathcal{P}|.$$

Thus, MAXPC is also NP-hard on the classes of graphs for which MINPC is NP-hard. Contrary to the case of MINPC, the (in)approximability aspect of this problem has received more attention in the literature. The basic idea of most of these algorithms is to find a 2-matching (a subgraph of maximum degree two) of maximum weight and remove (or add) edges from this subgraph to construct a path cover.

The first approximation algorithm for MAXPC was devised by Moran et al. [MNW90] in 1990. Their algorithm is based on the fact that, after removing an edge of minimum weight from each cycle, the total loss is at most $1/3$ of the weight of the initial solution, since every cycle has at least three edges. In Table 5.1, we summarize the results in this line of research.

KIND OF GRAPH	APPROX. FACTOR	RUNNING TIME
Undirected [MNW90]	$2/3$	$\mathcal{O}(m^2 \log m)$
Undirected, unit-weight [BK06]	$7/6$	$\mathcal{O}(n^9)$
Directed [MNW90]	$2/3$	$\mathcal{O}(nm + n^2 \log n)$
Directed [Vis92]	$7/12$	$\mathcal{O}(m^2 \log m)$

Table 5.1: Approximation algorithms for MAXPC.

As mentioned in the previous chapter, Vishwanathan [Vis92] showed an approximability relation between MAXPC and MINTSP-(1, 2). This relation is given by the following result.

Theorem 5.1 (Vishwanathan, 1992). *If there exists an α -approximation algorithm for MAXPC, then there is a $(2 - \alpha)$ -approximation algorithm for MINTSP-(1, 2).*

As Karpinsky & Schmied [KS13] showed an inapproximability threshold for MINTSP-(1, 2), stated below, combining it with the result above, we obtain an inapproximability result for MAXPC.

Theorem 5.2 (Karpinsky & Schmied, 2013). *Unless $P = NP$, there is no $(\frac{535}{534} - \epsilon)$ -approximation for MINTSP-(1, 2), for any $\epsilon > 0$.*

By Theorem 5.1 and Theorem 5.2, we have the following result.

Corollary 5.3. *Unless $P = NP$, there is no $(\frac{533}{534} + \epsilon)$ -approximation for MAXPC, for any $\epsilon > 0$.*

5.2 Nontrivial path covers

In this section we consider nontrivial path covers of optimum total weight. In contrast to the case where trivial paths are allowed, the minimization problem is not so simple. The two problems that we shall focus in this section are the following.

MINIMUM WEIGHT NONTRIVIAL PATH COVER (MINWNTPC): Given a graph G with nonnegative weight w_e for each edge e , find a nontrivial path cover $\mathcal{P}$ such that $w(\mathcal{P})$ is minimum, or answer that one does not exist.

Figure 5.1: *The T-reduction of a K_3 .*

MAXIMUM WEIGHT NONTRIVIAL PATH COVER (MAXWNTPC): Given a graph G with nonnegative weight w_e for each edge e , find a nontrivial path cover $\mathcal{P}$ such that $w(\mathcal{P})$ is maximum, or answer that one does not exist.

We recall that in Chapter 3 we have proved that we can decide in polynomial time whether a graph G admits a nontrivial path cover. Thus, we may assume that in the optimization problems MINWNTPC and MAXWNTPC the input graph G admits a nontrivial path cover.

As in the case of MINPC and MAXPC, we have that MINNTPC is equivalent to MAXWNTPC on unit-weight graphs and, therefore, MAXWNTPC is NP-hard. Now, we consider the MINWNTPC problem. In what follows, we show that MINWNTPC can be solved in polynomial time. This result is based on a reduction to a classical problem in combinatorial optimization. A **2-factor** of a graph G is a set of vertex-disjoint cycles that collectively spans $V(G)$.

Now, we describe a construction that we will use throughout this chapter. Let $G = (V, E)$ be a graph. Also, for each $e \in E$, let $w_e \in \mathbb{R}_+$ be the weight associated with $e \in E$. Let $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ be copies of G such that $v_i \in V_i$ is a copy of vertex $v \in V$. The **T-reduction** of G is the edge-weighted graph $G' = (V', E')$ obtained from G_1 and G_2 in the following way:

$$\begin{aligned} V' &= V_1 \cup V_2, \\ E' &= E_1 \cup E_2 \cup \{v_1v_2 : v \in V\}. \end{aligned}$$

We define weights on the edges of G' as follows:

$$w'_e = \begin{cases} 0, & \text{if } e = v_1v_2 \text{ for some } v \in V, \\ w_e, & \text{otherwise.} \end{cases}$$

In Figure 5.1 we show an example of this construction. Next, we show the following result.

Theorem 5.4. *MINWNTPC can be solved in polynomial time.*

Proof. Let $G = (V, E)$ be a graph that admits a nontrivial path cover. Also, let $G' = (V', E')$ be the T -reduction of G . Now, we prove that finding a minimum weight nontrivial path cover in G is equivalent to finding a 2-factor of minimum weight in G' . First, we show that G' has a 2-factor. Let $\mathcal{P}$ be a nontrivial path cover of G . Consider $\mathcal{P}_i$ as the copy of $\mathcal{P}$ in G_i , for $i = 1, 2$. To obtain a 2-factor of G' , we take each path in $\mathcal{P}_1$ and its copy in $\mathcal{P}_2$ and use the two edges linking its corresponding ends. In what follows, $\mathcal{P}^*$ (resp. $\mathcal{C}^*$) denotes a minimum weight nontrivial path cover (resp. 2-factor) of G (resp. G'). The previous construction shows that

$$w'(\mathcal{C}^*) \leq 2w(\mathcal{P}^*). \quad (5.1)$$

Now, we show that $w'(\mathcal{C}^*) \geq 2w(\mathcal{P}^*)$. Let $\mathcal{C}_1^*$ and $\mathcal{C}_2^*$ be the subgraphs induced by the edges of $\mathcal{C}^*$ in G_1 and G_2 , respectively. Observe that

$$w'(\mathcal{C}^*) = w'(\mathcal{C}_1^*) + w'(\mathcal{C}_2^*).$$

It is immediate that $w'(\mathcal{C}_1^*) = w'(\mathcal{C}_2^*)$. Otherwise, using the lightest of $\mathcal{C}_1^*$ and $\mathcal{C}_2^*$, we could construct a 2-factor $\hat{\mathcal{C}}$ with $w'(\hat{\mathcal{C}}) < w'(\mathcal{C}^*)$.

Since $\mathcal{C}^*$ is a 2-factor, and each vertex in G_1 is adjacent to only one vertex in G_2 , we have that $\mathcal{C}_1^*$ is a collection of cycles and nontrivial paths. Therefore, if we remove one edge from every cycle contained in $\mathcal{C}_1^*$, then we obtain a nontrivial path cover of G_1 , say $\mathcal{P}'$. Since the weights are nonnegative, this implies that

$$2w(\mathcal{P}^*) \leq 2w'(\mathcal{P}') \leq 2w'(\mathcal{C}_1^*) = w'(\mathcal{C}^*). \quad (5.2)$$

From (5.1) and (5.2), we have that $w(\mathcal{C}^*) = 2w(\mathcal{P}^*)$. Furthermore, when we remove one edge from each cycle of $\mathcal{C}_1^*$, we obtain a minimum weight nontrivial path cover of G . As we can find a 2-factor of minimum weight in polynomial time [Sch03], the result follows. $\square$

Since we can find a maximum weight 2-factor of a graph in polynomial time [Sch03], and using the T -reduction of a graph, we can show an approximation algorithm for MAXWNTPC.

Theorem 5.5. *There is a $(2/3)$ -approximation algorithm for MAXWNTPC.*

Proof. Let $G = (V, E)$ be a graph that admits a nontrivial path cover, and let $w_e \in \mathbb{R}_+$ be the weight associated with e , for every $e \in E$. Let $G' = (V', E')$ be the T -reduction of G with weight w'_e , for every $e \in E'$, with G_1 and G_2 being the two copies of G in G' , as in the proof of Theorem 5.4.

Let $\mathcal{P}^*$ be a nontrivial path cover of maximum weight in G . Also, let $\mathcal{C}$ be a maximum weight 2-factor of G' . By analogous arguments as those used in the proof of Theorem 5.4, we have that $2w(\mathcal{P}^*) \leq w'(\mathcal{C})$. Moreover, if $\mathcal{C}_1$ is the subgraph induced by the edges of $\mathcal{C}$ in G_1 , then

$$w(\mathcal{P}^*) \leq \frac{w'(\mathcal{C})}{2} = w'(\mathcal{C}_1) = w(\mathcal{C}_1).$$

Note that, if we remove an edge of minimum weight from every cycle in $\mathcal{C}_1$, we obtain a nontrivial

path cover of G , say $\mathcal{P}$, such that

$$w(\mathcal{P}) \geq \frac{2}{3}w(\mathcal{C}_1) \geq \frac{2}{3}w(\mathcal{P}^*).$$

□

We observe that, in the proof of Theorem 5.4 (resp. Theorem 5.5), the subgraph $\mathcal{C}_1^*$ (resp. $\mathcal{C}_1$) is a minimum (resp. maximum) weight $[1, 2]$ -factor of G . Therefore, we obtain the following.

Corollary 5.6. *Let G be a graph with weight $w_e \geq 0$ for each edge $e \in E(G)$. Then, we can find an optimum weight $[1, 2]$ -factor of G in polynomial time.*

5.3 Graphs with bounded treewidth

Many hard problems when restricted to graphs with bounded treewidth can be solved in polynomial time. Therefore, it is natural to ask whether this is the case for the path cover problems that we have considered in this text. As a matter of fact, these problems can be solved in linear time in this class of graphs. This result is more of theoretical interest, as large constant factors (exponential growth in terms of treewidth) are hidden in the $\mathcal{O}$ -notation.

Let $G = (V, E)$ be a graph. We show that the property “ G has a nontrivial path cover” can be formulated in monadic second order logic (MSOL). Roughly speaking, for graphs, in *second order logic* we can use quantifications on the vertices and the edges (or subsets of them), and the relation is the incidence relation of the graph. The *monadic second order logic* is a restriction of the second order logic in which quantifications are allowed only on first order variables and on unary relations. We will not elaborate more on this, but we refer the reader to [CE12], for a comprehensive work on this subject.

The following theorem, shown by Courcelle [Cou90], states an important algorithmic consequence for such properties.

Theorem 5.7 (Courcelle, 1990). *Every graph property definable in monadic second order logic can be decided in linear time on the class of graphs with bounded treewidth.*

We observe that the property “ G has a nontrivial path cover” is equivalent to “ G has a subset of edges $F \subseteq E$ that induces an acyclic $[1, 2]$ -factor”. We give below a formula φ that describes this property in MSOL. It is composed of other formulas that are expressed subsequently. All of them use the incidence relation **inc**, where $\text{inc}(a, b)$ is *true* if and only if b is an endpoint of a , or a is an endpoint of b .

$$\begin{aligned} \varphi \triangleq & \exists F \subseteq E, \\ & \forall v \in V [\deg 1(v, F) \vee \deg 2(v, F)] \wedge \\ & \forall Y \subseteq V [\text{conn}(Y, F) \Rightarrow \exists y \in Y [\deg 1(y, F)]], \end{aligned}$$

where $\text{conn}(Y, F)$, $\text{deg1}(v, F)$ and $\text{deg2}(v, F)$ are defined as follows:

$$\begin{aligned}\text{conn}(Y, F) &\triangleq \forall Z \subseteq V [(\exists u \in Y[u \in Z] \wedge \exists v \in Y[v \notin Z]) \Rightarrow \\ &\quad (\exists e \in F \exists u \in Y \exists v \in Y[\text{inc}(u, e) \wedge \text{inc}(v, e) \wedge u \in Z \wedge v \notin Z])]. \\ \text{deg1}(v, F) &\triangleq \exists e_1 \in F[\text{inc}(v, e_1) \wedge (\forall e_2 \in F[\text{inc}(v, e_2) \Rightarrow (e_2 = e_1)])]. \\ \text{deg2}(v, F) &\triangleq \exists e_1, e_2 \in F[(e_1 \neq e_2) \wedge \text{inc}(v, e_1) \wedge \text{inc}(v, e_2) \wedge \\ &\quad (\forall e_3 \in F[\text{inc}(v, e_3) \Rightarrow (e_3 = e_1) \vee (e_3 = e_2)])].\end{aligned}$$

Observe that $\text{conn}(Y, F)$ is true if, and only if, the set of vertices $Y \subseteq V$ induces a connected component in $G[F]$. On the other hand, $\text{deg1}(v, F)$ (resp. $\text{deg2}(v, F)$) is true if, and only if, the vertex $v \in V$ has degree one (resp. two) in $G[F]$. Since MINNTPC (resp. MAXNTPC) is equivalent to MAXWNTPC (resp. MINWNTPC) on unit-weight graphs, by a variant of Courcelle's Theorem (see [Rao07], Theorem 6), we obtain the following result.

Corollary 5.8. *MINNTPC, MAXNTPC, MINWNTPC and MAXWNTPC can be solved in linear time on the class of graphs with bounded treewidth.*

We note that the property “ G has a path cover” can also be formulated in MSOL. One can use an expression similar to the one we have exhibited above, and include the possibility of having vertices of degree zero (trivial components). The following formula ψ models this property.

$$\begin{aligned}\psi &\triangleq \exists F \subseteq E, \\ &\quad \forall v \in V[\text{deg0}(v, F) \vee \text{deg1}(v, F) \vee \text{deg2}(v, F)] \wedge \\ &\quad \forall Y \subseteq V[\text{conn}(Y, F) \Rightarrow \exists y \in Y[\text{deg0}(y, F) \vee \text{deg1}(y, F)]] ,\end{aligned}$$

where $\text{conn}(Y, F)$, $\text{deg1}(v, F)$ and $\text{deg2}(v, F)$ are defined as before and $\text{deg0}(v, F)$ is defined as follows:

$$\text{deg0}(v, F) \triangleq \forall e \in F[\neg \text{inc}(v, e)].$$

Since MINPC is equivalent to MAXPC on unit-weight graphs, we obtain the following.

Corollary 5.9. *MINPC and MAXPC can be solved in linear time on the class of graphs with bounded treewidth.*

5.4 The $[1, 2]$ -factor polytope

In Chapter 3 we showed a characterization for the existence of a nontrivial path cover in a graph, and observed that its existence is equivalent to the existence of a $[1, 2]$ -factor. Just to recall, a $[1, 2]$ -factor of a graph is a spanning subgraph in which every vertex has degree one or two. Let $G = (V, E)$ be a graph that admits a nontrivial path cover, and for each $e \in E$, let $w_e \in \mathbb{R}_+$ be the weight associated with e . Since each w_e is nonnegative, we can obtain a minimum weight nontrivial path cover of G from a minimum-weight $[1, 2]$ -factor of G , by removing one edge from

each cycle (if one exists). Observe that we can model the minimum weight $[1, 2]$ -factor problem by the following ILP, where the decision variable $x = (x_e)_{e \in E}$ indicates that $x_e = 1$ if and only if e is selected to belong to a $[1, 2]$ -factor of G .

$$\begin{aligned}
& \text{minimize} && \sum_{e \in E} w_e x_e \\
& \text{subject to} && x(\delta(v)) \leq 2, \quad \forall v \in V, \\
& && x(\delta(v)) \geq 1, \quad \forall v \in V, \\
& && x_e \in \{0, 1\}, \quad \forall e \in E.
\end{aligned} \tag{5.3}$$

Let us denote by $P(G)$ the polytope defined by the relaxation of (5.3), that is,

$$\begin{aligned}
x(\delta(v)) &\leq 2, \quad \forall v \in V, \\
x(\delta(v)) &\geq 1, \quad \forall v \in V, \\
x_e &\leq 1, \quad \forall e \in E, \\
x_e &\geq 0, \quad \forall e \in E.
\end{aligned} \tag{P(G)}$$

Now, define the *polytope of the $[1, 2]$ -factor of a graph G* ,

$$\mathcal{F}(G) = \mathbf{conv}\{\chi^F : F \text{ is a } [1, 2]\text{-factor of } G\}.$$

Since the integral vectors of $P(G)$ are precisely the incidence vectors of $[1, 2]$ -factors of G , we have that

$$\mathcal{F}(G) = \mathbf{conv}\{x \in P(G) : x \text{ is integral}\}.$$

Therefore,

$$\mathcal{F}(G) \subseteq P(G).$$

A natural question that arises now is whether the inclusion above holds with equality for every graph G ; and if not, whether there are classes of graphs for which equality holds.

In what follows, we show that $\mathcal{F}(G) = P(G)$ if G is a bipartite graph. Before we show this, we define the concept of totally unimodular matrices. We say that a matrix A is **totally unimodular** if every square submatrix of A has determinant 0, 1, or -1 . The following two results regarding totally unimodular matrices imply the integrality of $P(G)$ when G is bipartite (see [CCZ14]).

Theorem 5.10 (Hoffman, 1956). *Let A_G be the incidence matrix of a graph G . Then A_G is totally unimodular if and only if G is bipartite.*

Theorem 5.11 (Hoffman & Kruskal, 1956). *Let A be an $m \times n$ integral matrix. The polyhedron*

$$Q := \{x : c \leq Ax \leq d, \ell \leq x \leq u\}$$

is integral for all integer vectors c, d, ℓ, u if and only if A is totally unimodular.

By Theorem 5.10 and Theorem 5.11, the following result follows.

Figure 5.2: a) a graph G ; and b) a vertex x^1 of $P(G)$.

Corollary 5.12. *If G is a bipartite graph, then $\mathcal{F}(G) = P(G)$.*

Now, we show that there are graphs G for which $\mathcal{F}(G) \neq P(G)$; that is, $P(G)$ has non-integral vertices. Consider the graph $G = (V, E)$ and the vector $x^1 \in \mathbb{R}^E$ shown in Figure 5.2 b). Clearly, x^1 belongs to $P(G)$. We show that x^1 is a vertex of $P(G)$. Let $M \in \mathbb{R}^+$ and consider, for each edge $e \in E$, the weight w_e^1 defined as follows.

$$w_e^1 := \begin{cases} 1, & \text{if } x_e^1 > 0, \\ M, & \text{otherwise.} \end{cases}$$

Observe that if we choose M sufficiently large, any optimal solution of

$$\min \left\{ \sum_{e \in E} w_e^1 x_e : x \in P(G) \right\}$$

must have the same support as x^1 , implying that x^1 is its unique optimal solution. Thus, x^1 is a fractional vertex of $P(G)$, which implies that $P(G) \neq \mathcal{F}(G)$.

In what follows, we show three sets of inequalities that are valid for $\mathcal{F}(G)$, for any graph G . For each one of these inequalities we show a graph $\hat{G}$ and a fractional vertex of $P(\hat{G})$ that violates it. Furthermore, for each vertex of $P(\hat{G})$, we only show the edges on the support of that vertex.

Theorem 5.13. *Let $G = (V, E)$ be a graph. Then the inequality*

$$x(E[U]) + x(\delta(U)) \geq \left\lceil \frac{|U|}{2} \right\rceil, \quad \forall U \subseteq V, |U| \text{ odd}, \quad (5.4)$$

is valid for $\mathcal{F}(G)$.

Proof. First, observe that it suffices to show that the incidence vectors of $[1, 2]$ -factors of G satisfy (5.4). Let x be such a vector. For any $x \in \mathbb{R}^E$ and any $U \subseteq V$, we have that

$$\sum_{v \in U} x(\delta(v)) = 2x(E[U]) + x(\delta(U)).$$

Since $x \in \mathcal{F}(G)$, then $x(\delta(v)) \geq 1$ for all $v \in V$. Adding up this inequality for all $v \in U$, we obtain

$$2x(E[U]) + x(\delta(U)) \geq |U|.$$

Since x is nonnegative, $x(\delta(U)) \geq 0$. Thus, by adding this inequality to the inequality above, we obtain $2x(E[U]) + 2x(\delta(U)) \geq |U|$. Finally, as x is integer-valued, we have that

$$x(E[U]) + x(\delta(U)) \geq \left\lceil \frac{|U|}{2} \right\rceil.$$

□

Remark. There are graphs G for which the inequality (5.4) is not valid for $P(G)$ (it cuts fractional vertices of $P(G)$). Indeed, consider the graph G shown in Figure 5.2 a), and the vertex x^1 of $P(G)$ shown in Figure 5.2 b). Note that, if we choose U as the set of vertices induced by any of the two triangles in Figure 5.2 b), then

$$x^1(E[U]) + x^1(\delta(U)) = \frac{3}{2} < 2 = \left\lceil \frac{|U|}{2} \right\rceil.$$

Therefore, x^1 violates (5.4).

Inequality (5.4) is commonly known as the *edge cover constraint*, since it describes the polytope that arises from the incidence vectors of the edge covers of a graph [Sch03].

Next, we show the second set of valid inequalities we found for $\mathcal{F}(G)$. It is known as the *2-matching constraint* [Sch03].

Theorem 5.14. *Let $G = (V, E)$ be a graph. Then the inequality*

$$x(E[U]) + x(F) \leq |U| + \left\lfloor \frac{|F|}{2} \right\rfloor, \quad \forall U \subseteq V, F \subseteq \delta(U), F \text{ is a matching}, \quad (5.5)$$

is valid for $\mathcal{F}(G)$.

Proof. Let x be the incidence vector of a $[1, 2]$ -factor, say H , in G . Suppose by contradiction that there exist a set $U \subseteq V$ and a matching $F \subseteq \delta(U)$ such that

$$x(E[U]) + x(F) > |U| + \left\lfloor \frac{1}{2}|F| \right\rfloor,$$

and, from those, choose one that minimizes $|U| + |F|$. Note that, by the way we chose U and F , we have that

a) $x_f = 1$, for every $f \in F$, and

b) there is no cycle or path of H contained in U .

First, we show *a*). Observe that, if there is an edge $f' \in F$ such that $x'_{f'} = 0$, then by considering the matching $F' = F - f'$, we have

$$x(E[U]) + x(F') = x(E[U]) + x(F) > |U| + \left\lfloor \frac{1}{2}|F| \right\rfloor \geq |U| + \left\lfloor \frac{1}{2}|F'| \right\rfloor,$$

contradicting the way we chose U and F . Now, we show *b*). Let K be a path or cycle of H such that $V(K) \subseteq U$, and let $U' := U \setminus V(K)$. Since K is contained in U , by *a*), we have that the matching $F \subseteq \delta(U')$. Note that $x(E[U']) \geq x(E[U]) - |V(K)|$ and $|U'| = |U| - |V(K)|$, thus U' contradicts the minimality of U . Therefore, the subgraph $H[U]$ is a collection of disjoint paths. This implies that $x(E[U]) = |U| - k$, where k is the number of components of $H[U]$. Observe that by *a*), every edge in F must be incident to an end of a path in $H[U]$. Thus $|F| \leq 2k$, which implies that $k \geq \left\lceil \frac{1}{2}|F| \right\rceil$. Therefore,

$$\begin{aligned} x(E[U]) + x(F) &= |U| - k + |F| \\ &\leq |U| - \left\lceil \frac{1}{2}|F| \right\rceil + |F| \\ &= |U| + \left\lfloor \frac{1}{2}|F| \right\rfloor, \end{aligned}$$

a contradiction. □

Figure 5.3: *a*) a graph G and *b*) a vertex x^2 of $P(G)$ separated by inequality (5.5).

Remark. There are graphs G for which the inequality (5.5) cuts fractional vertices of $P(G)$. To show this, consider the graph G shown in Figure 5.3 *a*), and the vector x^2 of $P(G)$ shown in Figure 5.3 *b*). First, we show that x^2 is indeed a vertex of $P(G)$. Consider the weight w_e^2 , for each edge $e \in E$, defined as follows.

$$w_e^2 := \begin{cases} 1, & \text{if } x_e^2 = \frac{1}{2}, \\ 10, & \text{if } x_e^2 = 1, \\ -10, & \text{otherwise.} \end{cases}$$

Note that every optimal solution x^* of $\max\{\sum_{e \in E} w_e^2 x_e : x \in P(G)\}$ will have $x_e^* = 1$ (resp. $x_e^* = 0$) if $w_e^2 = 10$ (resp. $w_e^2 = -10$). Finally, for the three remaining edges, $x_e^* = \frac{1}{2}$. Therefore, x^2 is a

Figure 5.4: a) K_4 and b) a vertex of $P(K_4)$.

vertex of $P(G)$.

Now, consider the set $U \subseteq V$ shown in Figure 5.3 b) and the set F consisting of the three edges (in the support of x^2) that belong to $\delta(U)$. Then,

$$x^2(E[U]) + x^2(F) = \frac{13}{2} > 6 = |U| + \left\lfloor \frac{1}{2}|F| \right\rfloor.$$

Thus x^2 is separated by inequality (5.5).

Finally, we show the third valid inequality we found for $\mathcal{F}(G)$.

Theorem 5.15. *Let $G = (V, E)$ be a graph. Then the inequality*

$$x(E[V - u]) \geq \left\lceil \frac{|V| - 3}{2} \right\rceil, \quad \forall u \in V, \quad (5.6)$$

is valid for $\mathcal{F}(G)$.

Proof. Let x be the incidence vector of a $[1, 2]$ -factor of G . Note that, for any $x \in \mathbb{R}^E$ and any vertex $u \in V$, we have that

$$\sum_{v \in V - u} x(\delta(v)) = 2x(E[V - u]) + x(\delta(V - u)).$$

Since $x \in \mathcal{F}(G)$, then $x(\delta(v)) \geq 1$ for every $v \in V$. Adding up this inequality for all $v \in V - u$, we obtain

$$2x(E[V - u]) + x(\delta(V - u)) \geq |V| - 1.$$

Moreover, by the integrality of x and the fact that $x(\delta(V - u)) = x(\delta(u)) \leq 2$, we have that

$$x(E[V - u]) \geq \left\lceil \frac{|V| - 3}{2} \right\rceil.$$

□

Remark. There are graphs G for which inequality (5.6) cuts fractional vertices of $P(G)$. For this, consider the graph G shown in Figure 5.4 a), and the vector x^3 of $P(G)$ shown in Figure 5.4 b). First,

we show that x^3 is in fact a vertex of $P(G)$. Consider the weight w_e^3 defined for every edge $e \in E$, as follows.

$$w^3(e) := \begin{cases} 1, & \text{if } e = ab \text{ or } e = cd, \\ -2, & \text{if } e = ac \text{ or } e = ad, \\ 10, & \text{otherwise.} \end{cases}$$

Now, we show that x^3 is an optimal solution of $\min\{\sum_{e \in E} w^3(e)x_e : x \in P(G)\}$. Because of the high weights on the edges incident to b , except for ab , any optimal solution x^* must have $x_{ab}^* = 1$. Moreover, if we have $x_{ac}^* = 1$ or $x_{ad}^* = 1$, then the unique solution we can obtain has weight zero. On the other hand, $\sum_{e \in E} w_e^3 x_e^3 = -\frac{1}{2}$, which is minimum. Therefore, x^3 is a vertex of $P(G)$.

Finally, if we consider $u = a$, then

$$x^3(E[V - u]) = \frac{1}{2} < 1 = \left\lceil \frac{|V| - 3}{2} \right\rceil.$$

Therefore, x^3 is separated by inequality (5.6).

Unfortunately inequalities (5.4), (5.5), and (5.6) are not enough to describe $\mathcal{F}(G)$. In Figure 5.5, we show a fractional vertex of $P(G)$ that satisfies all the previous inequalities. Now, we show how

Figure 5.5: a) a graph G ; and b) a fractional vertex of $P(G)$.

to find, in polynomial time, if a feasible solution to $P(G)$ violates (5.4), (5.5), or (5.6).

Lemma 5.16. *Let $x \in P(G)$. Then, we can decide if x satisfies (5.4), (5.5), and (5.6) in polynomial time.*

Proof. Let $G = (V, E)$ be a graph. First, for inequality (5.6), we can precalculate $x(E[V])$ and subtract $x(\delta(u))$ to obtain $x(E[V - u])$, for $u \in V$. Thus, we can separate (5.6) in linear time. Now, consider inequality (5.4):

$$x(E[U]) + x(\delta(U)) \geq \left\lceil \frac{|U|}{2} \right\rceil \quad \forall U \subseteq V, |U| \text{ odd.}$$

From this inequality we obtain

$$\begin{aligned} x(E[U]) + x(\delta(U)) &\geq \frac{|U|+1}{2}, \\ 2x(E[U]) + 2x(\delta(U)) &\geq |U| + 1, \\ \sum_{v \in U} x(\delta(v)) + x(\delta(U)) &\geq |U| + 1, \\ \sum_{v \in U} (x(\delta(v)) - 1) + x(\delta(U)) &\geq 1. \end{aligned}$$

Note that $x(\delta(v)) - 1 \geq 0$ for every $v \in V$, since $x \in P(G)$. Consider the graph G^* obtained from G by adding a vertex s and linking it to every vertex $v \in V$. Also, consider the edge capacities c_e , for every $e \in E(G^*)$, defined as follows:

$$c_e := \begin{cases} x_e, & \text{if } e \in E, \\ x(\delta(v)) - 1, & \text{if } e = sv, v \in V. \end{cases}$$

Then, the expression

$$\sum_{v \in U} (x(\delta(v)) - 1) + x(\delta(U))$$

can be seen as the value of the cut induced by the set $U \subseteq V$ in G^* . Therefore, we have reduced our problem to finding a minimum odd cut, not containing vertex s , in G^* . To find this minimum odd cut in polynomial time, we can use the Gomory-Hu tree [LRT08].

Finally, consider the inequality (5.5).

$$x(E[U]) + x(F) \leq |U| + \left\lfloor \frac{|F|}{2} \right\rfloor, \quad \forall U \subseteq V, F \subseteq \delta(U), F \text{ is a matching},$$

and the inequalities that follow

$$\begin{aligned} 2x(E[U]) + 2x(F) &\leq 2|U| + |F| - 1, \\ \sum_{v \in U} x(\delta(v)) - x(\delta(U)) + 2x(F) &\leq 2|U| + |F| - 1, \\ 2|U| - \sum_{v \in U} x(\delta(v)) + |F| - x(F) + x(\delta(U)) - x(F) &\geq 1, \\ \sum_{v \in U} (2 - x(\delta(v))) + \sum_{f \in F} (1 - x_f) + x(\delta(U) \setminus F) &\geq 1. \end{aligned}$$

As in the previous case, we can construct the graph G^* from G , but with the following edge capacities:

$$c_e := \begin{cases} x_e, & \text{if } e \in E, \\ 2 - x(\delta(v)), & \text{if } e = sv, v \in V. \end{cases}$$

Although the expression

$$\sum_{v \in U} (2 - x(\delta(v))) + \sum_{f \in F} (1 - x_f) + x(\delta(U) \setminus F)$$

does not correspond exactly to a minimum cut in G^* , we can still decide if x satisfies (5.5), in polynomial time, using the Gomory-Hu tree [LRT08]. $\square$

In our studies we were seeking a complete linear description of $\mathcal{F}(G)$. As we have seen, there are still fractional vertices of $P(G)$ that need to be cut off in order to obtain our desired result. But, suppose that we have a candidate polytope, say $Q(G)$, that contains $\mathcal{F}(G)$. Next, we show a method that could be used to prove that $Q(G) = \mathcal{F}(G)$.

Let $G = (V, E)$ be a graph, and let $G' = (V', E')$ be the T -reduction of G , defined on page 49. For any vector x in $Q(G)$, we define the vector $x' \in \mathbb{R}^{E'}$ as follows

$$x'_e = \begin{cases} x_{uv}, & \text{if } e = u_1v_1 \text{ or } e = u_2v_2, \\ 2 - x(\delta(v)), & \text{if } e = v_1v_2. \end{cases}$$

In what follows we will show a relation between $Q(G)$ and the polytope $T(G')$ of the 2-factors of G' defined as [Sch03]:

$$\begin{aligned} x'(\delta(v)) &= 2, & \forall v' \in V', \\ x'(\delta(U)) &\geq 2x'(F) - |F| + 1, & \forall U \subseteq V', F \subseteq \delta(U), F \text{ matching, } |F| \text{ odd} \\ x'_e &\leq 1, & \forall e \in E', \\ x'_e &\geq 0, & \forall e \in E'. \end{aligned} \quad (T(G'))$$

Lemma 5.17. *Let x be any vector in $Q(G)$. If $x' \in T(G')$, then the $[1, 2]$ -factor polytope $\mathcal{F}(G)$ is determined by $Q(G)$.*

Proof. Suppose by contradiction that $Q(G)$ does not define the $[1, 2]$ -factor polytope of G . Then, there is a fractional vertex, say y , of $Q(G)$. Let $y' \in T(G')$ be the vector defined from y . Since $T(G')$ determines the polytope of the 2-factors of G' , then y' can be expressed as a convex combination

$$y' = \sum_{i=1}^k \alpha_i \chi^i,$$

where χ^i is the incidence vector of a 2-factor of G' , for $i = 1, \dots, k$. Note that, by the definition of G' , when we restrict χ^i to the edges in E_1 , the resulting vector is the incidence vector of a $[1, 2]$ -factor in G_1 . Let χ_1^i be the restriction of χ^i to G_1 , for $i = 1, \dots, k$. Then

$$y'_1 = \sum_{i=1}^k \alpha_i \chi_1^i,$$

where y'_1 is the restriction of y' to G_1 . Since G_1 is a copy of G , we have that χ_1^i is also an incidence vector of a $[1, 2]$ -factor of G . Therefore, y is a convex combination of vectors in $Q(G)$, a contradiction. $\square$

Lemma 5.17 tells us that, to prove that $Q(G)$ determines the polytope of $[1, 2]$ -factors of G , we just need to show that if $x \in Q(G)$, then $x' \in T(G')$.

5.5 Final remarks

The results presented in Section 5.4 were obtained more recently. During this process, we used PANDA [LR15], a program that *a)* given a set of vectors, generates a set of inequalities that describes the convex hull of those vectors; and *b)* given a set of inequalities, generates all the vertices of the polytope described by those inequalities. Since this program runs in exponential time, we used it to generate the complete description of the $[1, 2]$ -polytope for some small cubic graphs. Using this program, we discovered inequalities (5.4) and (5.6).

Using Gurobi [Gur19], we implemented the linear formulation $P(G)$ along with the separation routines for inequalities (5.4) and (5.5). We ran several experiments on random graphs and cubic graphs using uniform weight over the edges. For all these experiments, the optimal solutions we obtained were integral. Let $G = (V, E)$ be a graph and let $\mathbb{1}^E$ denote the vector in $\mathbb{R}^E$ whose entries are all equal to one. Our experiments led us to the following question.

Are inequalities (5.4) and (5.5) enough to cut all the fractional vertices of $P(G)$ reachable in the direction $\mathbb{1}^E$?

Besides random graphs, we tested our implementation on all the cubic graphs of order n , $4 \leq n \leq 22$. We used the database “The House of Graphs” [BCGM13] to obtain the description of those graphs.

Chapter 6

Final considerations

In this thesis we addressed the path cover problem (cardinality and weighted versions) forbidding trivial paths. We summarize, in Table 6.1, the complexity and (in)approximability status of the problems studied in this text. We distinguish with a shaded cell the results that we obtained. In what follows, we recall other results that we obtained, present some concluding remarks, and mention lines for future research.

In Chapter 3, we were mainly concerned with the existence of a nontrivial path cover. We noticed the equivalence between the existence of such cover and the existence of a $[1, 2]$ -factor. Although this relation already gives us an algorithm that solves this problem, we followed another approach. We focused on the graphs that admit a path cover by paths of length one and two. This approach ended up being fruitful: we obtained a new characterization (Theorem 3.7) of graphs that admit a nontrivial path cover ($[1, 2]$ -factor), and a polynomial-time algorithm that either finds a solution to MAXNTPC or a negative certificate for its existence.

After that, we focused on MINPC and MINNTPC, especially their relation with respect to their computational complexity and approximability. We designed a polynomial-time algorithm that transforms any path cover into one with the same number of paths, but without trivial paths. The latter implies that MINPC and MINNTPC are, essentially, equivalent: they have the same computational complexity and approximability characteristics in the class of graphs that admit

Cardinality version (number of paths)			
	Comp. complexity	Best approx. factor	Inapprox. threshold
MINPC	NP-hard	?	$2 - \epsilon$
MINNTPC	NP-hard	?	$2 - \epsilon$
MAXNTPC	P	-	-
Weighted version (weight of the paths)			
	Comp. complexity	Best approx. factor	Inapprox. threshold
MAXPC	NP-hard	$2/3$	$533/534 + \epsilon$
MAXWNTPC	NP-hard	$2/3$	?
MINWNTPC	P	-	-

Table 6.1: Computational complexity and approximability of the problems we studied.

a nontrivial path cover. Although this result (the transformation) already gives us polynomial-time algorithms to solve MINNTPC in several classes of graphs, we designed a direct linear-time algorithm for this problem when the input graph is a tree. As shown in Figure 4.1, there are still many classes of graphs for which the computational complexity of MINPC (MINNTPC) is still unknown. It would be interesting to investigate MINPC on those classes.

Besides the equivalence of MINPC and MINNTPC in terms of approximability, we showed a relation between the approximation factor of MINPC and MINTSP-(1, 2). In this direction, a very challenging open problem, in our view, is to find an approximation algorithm for MINPC (MINNTPC). Regarding this problem, the optimal value of the relaxation of the integer formulation presented in Section 4.5 gives us a lower bound for the optimal value of MINPC. However, to design an approximation algorithm we need a better understanding of the fractional solutions of this relaxation. In what follows we comment more on this.

In the direction of exact algorithms, we presented two integer formulations, one for MINPC and the other for MINNTPC. Computational experiments have shown that both formulations perform well on random graphs. Using Gurobi [Gur19], we were able to solve most of the instances of up to 900 vertices in less than thirty minutes. Since MINPC has applications in network design and relations with other problems (like $L(2, 1)$ -labelling), we consider that the study of those formulations is not only of theoretical, but also of practical interest. It would be interesting to understand the structure of the graphs that are difficult for those formulations to solve. Possibly, the study of their linear relaxation, especially the study of the “structure” of the subgraphs induced by the support of their fractional vertices may help in the design of an approximation algorithm.

Finally, we studied the weighted version of the path cover problem. When we consider nontrivial path covers, we showed that MINWNTPC can be solved in polynomial time, while MAXWNTPC is NP-hard. Regarding MAXWNTPC, we showed a $(2/3)$ -approximation algorithm: a similar approximation factor known for MAXPC. One open problem regarding MAXPC and MAXNTPC is whether there exists a relation between the (in)approximability of those problems, as in the case of MINPC and MINNTPC. In Section 5.4, we studied the convex hull $\mathcal{F}(G)$ of the incidence vectors of the $[1, 2]$ -factors of a graph G . Mainly, we investigated the linear relaxation $P(G)$ of an integer formulation for $\mathcal{F}(G)$. We showed that the vertices of $P(G)$ are integral when G is bipartite. In the general case $P(G)$ may have fractional vertices. We showed three classes of inequalities that are valid for $\mathcal{F}(G)$ and may cut off fractional vertices of $P(G)$. Unfortunately we could not show a complete linear description for $\mathcal{F}(G)$. The fact that we can find an optimal weight $[1, 2]$ -factor in polynomial time makes us believe that such description exists. This is the case for several problems like matching, 2-matching, 2-factor, edge cover, etc. The search for more valid (facet-inducing) inequalities seems an interesting line for future research. We plan to investigate further the polytope $\mathcal{F}(G)$, hoping to find its complete linear description or some other interesting feature.

The results shown in Chapter 3, Chapter 4 (except for Section 4.5) and Section 5.2 were presented at the “44th International Workshop on Graph-Theoretic Concepts in Computer Science” (WG 2018) [GW18]. A complete version of that work has been submitted to an international journal [GW19]. Moreover, during the course of our research, we had the opportunity to collaborate

with other researchers. The results obtained from that collaboration were presented at the “*IX Latin and American Algorithms, Graphs and Optimization Symposium*” (LAGOS’ 17) [CFG⁺17] and a full version of that work has been submitted to an international journal [CFG⁺19].

Bibliography

- [ALS91] S. Arnborg, J. Lagergren and D. Seese. Easy problems for tree-decomposable graphs. *J. Algorithms*, 12(2):308–340, 1991. [26](#)
- [Ans85] R. Anstee. An algorithmic proof of Tutte’s f -factor theorem. *J. Algorithms*, 6(1):112–131, 1985. [14](#), [20](#)
- [AP90] S. Arikati and C. Pandu Rangan. Linear algorithm for optimal path cover problem on interval graphs. *Inform. Process. Lett.*, 35(3):149–153, 1990. [3](#), [26](#)
- [BA99] A. Barabási and R. Albert. Emergence of scaling in random networks. *Science*, 286(5439):509–512, 1999. [37](#), [38](#)
- [BCGM13] G. Brinkmann, K. Coolsaet, J. Goedgebeur and H. Mélot. House of Graphs: a database of interesting graphs. *Discrete Appl. Math.*, 161(1-2):311–314, 2013. [61](#)
- [BCM74] F. Boesch, S. Chen and J. McHugh. On covering the points of a graph with point disjoint paths. pages 201–212. *Lecture Notes in Math.*, Vol. 406, 1974. [1](#), [3](#)
- [BG77] F. Boesch and J. Gimpel. Covering the points of a digraph with point-disjoint paths and its application to code optimization. *J. ACM*, 24(2):192–198, 1977. [1](#), [26](#)
- [BK06] P. Berman and M. Karpinski. 8/7-approximation algorithm for (1, 2)-TSP. In *Proceedings of the Seventeenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 641–648. ACM, New York, 2006. [48](#)
- [BM08] J. Bondy and U. Murty. *Graph theory*, volume 244 of *Graduate Texts in Mathematics*. Springer, New York, 2008. [7](#), [9](#)
- [Bol98] B. Bollobás. *Modern graph theory*, volume 184 of *Graduate Texts in Mathematics*. Springer-Verlag, New York, 1998. [37](#)
- [CCD⁺01] M. Carvalho, M. Cerioli, R. Dahab, P. Feofiloff, C. Fernandes, C. Ferreira, K. Guimarães, F. Miyazawa, J. Pina, J. Soares and Y. Wakabayashi. *Uma Introdução Sucinta a Algoritmos de Aproximação*. First edition, 2001. [7](#), [12](#)
- [CCZ14] M. Conforti, G. Cornuéjols and G. Zambelli. *Integer programming*, volume 271 of *Graduate Texts in Mathematics*. Springer, Cham, 2014. [11](#), [53](#)
- [CDH13] D. Corneil, B. Dalton and M. Habib. LDFS-based certifying algorithm for the minimum path cover problem on cocomparability graphs. *SIAM J. Comput.*, 42(3):792–807, 2013. [3](#), [26](#)

- [CE12] B. Courcelle and J. Engelfriet. *Graph structure and monadic second-order logic*, volume 138 of *Encyclopedia of Mathematics and its Applications*. Cambridge University Press, Cambridge, 2012. A language-theoretic approach, With a foreword by Maurice Nivat. [51](#)
- [CFG⁺17] M. Cerioli, C. Fernandes, R. Gómez, J. Gutiérrez and P. Lima. Transversals of longest paths. In *LAGOS'17—IX Latin and American Algorithms, Graphs and Optimization*, volume 62 of *Electron. Notes Discrete Math.*, pages 135–140. Elsevier Sci. B. V., Amsterdam, 2017. [65](#)
- [CFG⁺19] M. Cerioli, C. Fernandes, R. Gómez, J. Gutiérrez and P. Lima. Transversals of longest paths. *Discrete Math.*, 2019. Accepted for publication. [65](#)
- [Chv83] V. Chvátal. *Linear programming*. A Series of Books in the Mathematical Sciences. W. H. Freeman and Company, New York, 1983. [11](#)
- [Cou90] B. Courcelle. The monadic second-order logic of graphs. I. Recognizable sets of finite graphs. *Inform. and Comput.*, 85(1):12–75, 1990. [51](#)
- [Dam89] P. Damaschke. The Hamiltonian circuit problem for circle graphs is NP-complete. *Inform. Process. Lett.*, 32(1):1–2, 1989. [1](#)
- [Die18] R. Diestel. *Graph theory*, volume 173 of *Graduate Texts in Mathematics*. Springer, Berlin, fifth edition, 2018. Paperback edition of [MR3644391]. [7](#), [9](#)
- [DM58] A. Dulmage and N. Mendelsohn. Coverings of bipartite graphs. *Canad. J. Math.*, 10:517–534, 1958. [18](#)
- [ER59] P. Erdős and A. Rényi. On random graphs. I. *Publ. Math. Debrecen*, 6:290–297, 1959. [37](#)
- [FR02] D. Franzblau and A. Raychaudhuri. Optimal Hamiltonian completions and path covers for trees, and a reduction to maximum flow. *ANZIAM J.*, 44(2):193–204, 2002. [26](#)
- [GJ79] M. Garey and D. Johnson. *Computers and intractability*. W. H. Freeman and Co., San Francisco, Calif., 1979. A guide to the theory of NP-completeness, A Series of Books in the Mathematical Sciences. [1](#), [10](#)
- [GLS93] M. Grötschel, L. Lovász and A. Schrijver. *Geometric algorithms and combinatorial optimization*, volume 2 of *Algorithms and Combinatorics*. Springer-Verlag, Berlin, second edition, 1993. [11](#)
- [GMW94] J. Georges, D. Mauro and M. Whittlesey. Relating path coverings to vertex labellings with a condition at distance two. *Discrete Math.*, 135(1):103 – 111, 1994. [3](#)
- [Gol04] M. Golumbic. *Algorithmic graph theory and perfect graphs*, volume 57 of *Annals of Discrete Mathematics*. Elsevier Science B.V., Amsterdam, second edition, 2004. With a foreword by Claude Berge. [1](#)
- [Gur19] Gurobi Optimization, LLC. Gurobi Optimizer Reference Manual, 2019. [37](#), [61](#), [64](#)
- [GW18] R. Gómez and Y. Wakabayashi. Covering a graph with nontrivial vertex-disjoint paths: Existence and optimization. In *Graph-Theoretic Concepts in Computer Science - 44th International Workshop, WG 2018, Cottbus, Germany, June 27-29, 2018. Lecture Notes in Computer Science, vol 11159, pp. 228–238*, pages 228–238, 2018. [64](#)

- [GW19] R. Gómez and Y. Wakabayashi. Nontrivial path covers of graphs: existence, minimization and maximization. *J. Comb. Optim.*, 2019. Accepted for publication. [64](#)
- [HC06] R. Hung and M. Chang. Solving the path cover problem on circular-arc graphs by using an approximation algorithm. *Discrete Appl. Math.*, 154(1):76–105, 2006. [3](#), [26](#)
- [HC07] R. Hung and M. Chang. Finding a minimum path cover of a distance-hereditary graph in polynomial time. *Discrete Appl. Math.*, 155(17):2242–2256, 2007. [26](#)
- [HHKL90] K. Heinrich, P. Hell, D. Kirkpatrick and G. Liu. A simple existence criterion for $(g < f)$ -factors. *Discrete Math.*, 85(3):313–317, 1990. [14](#)
- [HK73] J. Hopcroft and R. Karp. An $n^{5/2}$ algorithm for maximum matchings in bipartite graphs. *SIAM J. Comput.*, 2:225–231, 1973. [23](#)
- [HSS08] A. Hagberg, D. Schult and P. Swart. Exploring network structure, dynamics, and function using networkx. In G. Varoquaux, T. Vaught and J. Millman, editors, *Proceedings of the 7th Python in Science Conference*, pages 11 – 15, 2008. [37](#)
- [HW17] M. Henning and K. Wash. Matchings, path covers and domination. *Discrete Math.*, 340(1):3207–3216, 2017. [4](#)
- [Kar84] N. Karmarkar. A new polynomial-time algorithm for linear programming. *Combinatorica*, 4(4):373–395, 1984. [10](#)
- [Kha79] L. Khachijan. A polynomial algorithm in linear programming. *Dokl. Akad. Nauk SSSR*, 244(5):1093–1096, 1979. [10](#)
- [KKK04] M. Kano, G. Katona and Z. Király. Packing paths of length at least two. *Discrete Math.*, 283(1-3):129–135, 2004. [23](#)
- [KS83] M. Kano and A. Saito. $[a, b]$ -factors of graphs. *Discrete Math.*, 47(1):113–116, 1983. [13](#), [14](#)
- [KS13] M. Karpinski and R. Schmied. Improved inapproximability results for the shortest superstring and related problems. In *Proceedings of the Nineteenth Computing: The Australasian Theory Symposium - Volume 141*, CATS '13, pages 27–36, Darlinghurst, Australia, Australia, 2013. Australian Computer Society, Inc. [48](#)
- [KV12] B. Korte and J. Vygen. *Combinatorial optimization*, volume 21 of *Algorithms and Combinatorics*. Springer, Heidelberg, fifth edition, 2012. Theory and algorithms. [22](#)
- [LM98] Y. Li and C. Mao-Cheng. A degree condition for a graph to have $[a, b]$ -factors. *J. Graph Theory*, 27(1):1–6, 1998. [14](#)
- [LOP95] R. Lin, S. Olariu and G. Pruesse. An optimal path cover algorithm for cographs. *Comput. Math. Appl.*, 30(8):75–83, 1995. [26](#)
- [Lov70] L. Lovász. Subgraphs with prescribed valencies. *J. Combinatorial Theory*, 8:391–416, 1970. [14](#)
- [LP86] L. Lovász and M. Plummer. *Matching theory*, volume 121 of *North-Holland Mathematics Studies*. North-Holland, Amsterdam, 1986. [16](#), [21](#)

- [LR15] S. Lörwald and G. Reinelt. PANDA: a software for polyhedral transformations. *EURO J. Comput. Optim.*, 3(4):297–308, 2015. [61](#)
- [LRT08] A. Letchford, G. Reinelt and D. Theis. Odd minimum cut sets and b -matchings revisited. *SIAM J. Discrete Math.*, 22(4):1480–1487, 2008. [59](#), [60](#)
- [LV78] M. Las Vergnas. An extension of Tutte’s 1-factor theorem. *Discrete Math.*, 23(3):241–255, 1978. [14](#), [20](#)
- [Mö96] H. Müller. Hamiltonian circuits in chordal bipartite graphs. *Discrete Math.*, 156(1-3):291–298, 1996. [1](#)
- [MM09] C. Magnant and D. Martin. A note on the path cover number of regular graphs. *Australas. J. Combin.*, 43:211–217, 2009. [3](#)
- [MNW90] S. Moran, I. Newman and Y. Wolfstahl. Approximation algorithms for covering a graph by vertex-disjoint paths of maximum total weight. *Networks*, 20(1):55–64, 1990. [48](#)
- [MW91] S. Moran and Y. Wolfstahl. Optimal covering of cacti by vertex-disjoint paths. *Theoret. Comput. Sci.*, 84(2, Algorithms Automat. Complexity Games):179–197, 1991. [26](#)
- [MWY16] C. Magnant, H. Wang and S. Yuan. Path partitions of almost regular graphs. *Australas. J. Combin.*, 64:334–340, 2016. [3](#)
- [Nar89] G. Narasimhan. A note on the Hamiltonian circuit problem on directed path graphs. *Inform. Process. Lett.*, 32(4):167–170, 1989. [1](#)
- [PW87] S. Pinter and Y. Wolfstahl. On mapping processes to processors in distributed systems. *Internat. J. Parallel Programming*, 16(1):1–15, 1987. [1](#)
- [PY93] C. Papadimitriou and M. Yannakakis. The traveling salesman problem with distances one and two. *Math. Oper. Res.*, 18(1):1–11, 1993. [34](#)
- [Rao07] M. Rao. MSOL partitioning problems on graphs of bounded treewidth and clique-width. *Theoret. Comput. Sci.*, 377(1-3):260–267, 2007. [52](#)
- [Ree96] B. Reed. Paths, stars and the number three. *Combin. Probab. Comput.*, 5(3):277–295, 1996. [3](#)
- [Sch03] A. Schrijver. *Combinatorial optimization. Polyhedra and efficiency. Vol. A*, volume 24 of *Algorithms and Combinatorics*. Springer-Verlag, Berlin, 2003. Paths, flows, matchings, Chapters 1–38. [7](#), [11](#), [50](#), [55](#), [60](#)
- [SSSP93] R. Srikant, R. Sundaram, K. Sher Singh and C. Pandu Rangan. Optimal path cover problem on block graphs and bipartite permutation graphs. *Theoret. Comput. Sci.*, 115(2):351–357, 1993. [3](#), [26](#)
- [Tut52] W. Tutte. The factors of graphs. *Canadian J. Math.*, 4:314–328, 1952. [14](#)
- [Vaz01] V. Vazirani. *Approximation algorithms*. Springer-Verlag, Berlin, 2001. [12](#)
- [Vaz12] V. Vazirani. A Simplification of the MV Matching Algorithm and its Proof. *arXiv e-prints*, page arXiv:1210.4594, Oct 2012. [23](#)

- [Vis92] S. Vishwanathan. An approximation algorithm for the asymmetric travelling salesman problem with distances one and two. *Inform. Process. Lett.*, 44(6):297–302, 1992. [34](#), [48](#)
- [Wan94] H. Wang. Path factors of bipartite graphs. *J. Graph Theory*, 18(2):161–167, 1994. [23](#)
- [WS11] D. Williamson and D. Shmoys. *The design of approximation algorithms*. Cambridge University Press, Cambridge, 2011. [12](#)
- [Yu18] G. Yu. Covering 2-connected 3-regular graphs with disjoint paths. *J. Graph Theory*, 88(3):385–401, 2018. [3](#)
- [ZWZ17] S. Zhou, J. Wu and T. Zhang. The existence of $P_{\geq 3}$ -factor covered graphs. *Discuss. Math. Graph Theory*, 37(4):1055–1065, 2017. [23](#)
- [ZZ09] H. Zhang and S. Zhou. Characterizations for $\mathcal{P}_{\geq 2}$ -factor and $\mathcal{P}_{\geq 3}$ -factor covered graphs. *Discrete Math.*, 309(8):2067–2076, 2009. [23](#)