Graph Packing Problems

by

Sarah Pantel
B.Sc. (Major), University of Manitoba, 1994

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
Master of Science
in the Department
of
Mathematics and Statistics

© Sarah Pantel 1999
SIMON FRASER UNIVERSITY
October 1999

All rights reserved. This work may not be
reproduced in whole or in part, by photocopy
or other means, without the permission of the author.
The author has granted a non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of this thesis in microform, paper or electronic formats.

The author retains ownership of the copyright in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author’s permission.

L’auteur a accordé une licence non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de cette thèse sous la forme de microfiche/film, de reproduction sur papier ou sur format électronique.

L’auteur conserve la propriété du droit d’auteur qui protège cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

0-612-51442-0
Abstract

The problem of covering the vertices of a given undirected graph with a maximum number of disjoint copies of the complete graph on two vertices, $K_2$, is called the maximum matching problem. Due to the fruitful results of matching theory, such as the polynomial algorithms for finding the maximum matching of a given graph, it was hypothesized that similar results might hold for the analogous problem using a graph other than $K_2$. The problem of covering a graph with copies of graphs other than only $K_2$ is called the graph packing problem. Let $\mathcal{G}$ be a family of graphs. Formally, a $\mathcal{G}$-packing of a graph $H$ is a set of vertex disjoint subgraphs $\{G_1, \ldots, G_k\}$ of $H$ such that each $G_i$ is isomorphic to some graph in $\mathcal{G}$. The graph packing problem has been studied for a variety of families $\mathcal{G}$. Although the positive results are not as abundant as for the matching problem, many polynomial results have been found for graph packing problems on undirected graphs.

The undirected graph packing problem has now been studied quite extensively. However, this is not the case for the directed packing problem. In this thesis, we study the directed graph packing problem for families of directed paths, directed cycles and directed stars. The main new result of the thesis is a Berge-like characterization of maximum packings for the directed graph packing problem with the family $\{\vec{P}_1, \vec{P}_2\}$. 
Acknowledgments

Firstly, I would like to acknowledge my senior supervisor, Dr. Pavol Hell, for his direction and advice on this thesis. I would also like to thank my other supervisors Dr. Brian Alspach and Dr. Rick Brewster for serving on the committee. I am very grateful to Dr. Brewster for his invaluable help with the editing of this thesis, with the proof of the main result of Chapter 3 as well as for his financial support. I would also like to thank N.S.E.R.C. for the financial support of a PGS A award throughout most of this degree and the Department of Mathematics and Statistics for various TA positions.

I would like to thank my parents for their encouragement and prayers and my parents-in-law for their support. Lastly, but most importantly, I'd like to thank my husband, Christian, for constant encouragement, understanding and hours of proof-reading throughout the preparation of this thesis and the Lord Jesus Christ for helping me to persevere with my studies and for giving me strength and the right perspective.

"Give me, O Lord, a steadfast heart, which no unworthy affection may drag downwards; give me an unconquered heart, which no tribulation can wear out; give me an upright heart, which no unworthy purpose may tempt aside. Bestow on me also, O Lord my God, understanding to know you, diligence to seek you, wisdom to find you, and a faithfulness that may finally embrace you, through Jesus Christ our Lord."

Thomas Aquinas

"I said, ‘You are my servant’; I have chosen you and have not rejected you. So do not fear, for I am with you. Do not be dismayed, for I am your God. I will strengthen you and help you; I will uphold you with my righteous right hand."

Isaiah 41:9-10 (NIV)
Dedication

To my husband Christian
Contents

Abstract ................................................................. iii
Acknowledgments ......................................................... iv
Dedication ............................................................... v
List of Tables .......................................................... viii
List of Figures ......................................................... ix

1 Introduction .......................................................... 1
  1.1 Definitions and Notation ........................................ 1
  1.2 Background ....................................................... 4
  1.3 The Problem ..................................................... 5
  1.4 Complexity Theory ............................................. 6

2 Survey of the General Graph Packing Problem ..................... 9
  2.1 Families of Complete Graphs .................................. 9
      2.1.1 \(\{K_2, F_i\}\)-Packings ............................. 18
  2.2 Families of Complete Bipartite Graphs ....................... 20
  2.3 Families of Paths .............................................. 23
  2.4 Families of Cycles ............................................. 28
  2.5 Families of Mixed Graphs .................................... 28
  2.6 Applications .................................................. 29

3 Directed Graph Packings .......................................... 31
  3.1 Definitions and Notation ...................................... 31
  3.2 Packing with Directed Paths .................................. 32
      3.2.1 One-Element Families ................................... 32
  3.3 Main Theorem .................................................. 42
  3.4 Packing with Directed Cycles ................................ 59
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.5 Packing with Directed Stars</td>
<td>62</td>
</tr>
<tr>
<td>Conclusion</td>
<td>68</td>
</tr>
<tr>
<td>Bibliography</td>
<td>73</td>
</tr>
</tbody>
</table>
List of Tables

4.1 Complexity of the packing problem for families of cliques ............... 69
4.2 Complexity of the packing problem for families of complete bipartite graphs 70
4.3 Complexity of the packing problem for families of paths .................. 70
4.4 Complexity of the packing problem for families of cycles ................. 70
4.5 Complexity of the packing problem for mixed families ..................... 71
4.6 Complexity of the packing problem in the directed case .................. 71
List of Figures

1.1 $H = (V(H), E(H))$ .................................................. 2
1.2 Examples of commonly used graphs .................................. 2
1.3 A perfect matching of $C_6$ ............................................. 3

2.1 Gadget for the $\{K_3\}$-factor problem ................................. 10
2.2 A $\{K_3\}$-factor containing the connector vertices ................. 11
2.3 A $\{K_3\}$-factor excluding the connector vertices .................. 12
2.4 Augmenting configurations for the $\{K_2, K_3\}$-packing problem .... 14
2.5 Vertex $v$ covered by a $G'$-triangle .................................. 16
2.6 Vertices $w$ and $z$ covered by $G$-edges ............................... 16
2.7 Augmenting path between $x$ and $y$ .................................... 17
2.8 The kite for $G = \{K_2, C_3\}$ ........................................ 19
2.9 Augmenting star configurations ........................................ 22
2.10 Augmented star configurations ......................................... 22
2.11 Gadget for the $\{P_2\}$-factor problem ................................. 24
2.12 A $\{P_2\}$-factor containing the connector vertices ................. 25
2.13 A $\{P_2\}$-factor excluding the connector vertices .................. 25
2.14 A $\{P_2\}$-Packing of a Square Graph ............................... 26
2.15 A $\{P_1, P_2\}$-Factor of a Triangle Graph ........................... 27
2.16 Maximum isolation of vertices incident with $x$ ....................... 27

3.1 Gadget for the $\{\tilde{P}_2\}$-factor problem ............................ 33
3.2 A $\{\tilde{P}_2\}$-factor containing the connector vertices ............... 33
3.3 A $\{\tilde{P}_2\}$-factor excluding the connector vertices ................ 34
3.4 Colouring of a $\tilde{P}_4$ used in $\text{COL-}\{\tilde{P}_4\}$ ...................... 36
3.5 Gadget for the COL-{\(P_4\)}-factor problem ........................................ 37
3.6 A COL-{\(P_4\)}-factor of the gadget including the connectors .......... 37
3.7 A COL-{\(P_4\)}-factor of the gadget excluding the connectors .......... 38
3.8 Placement of \(P_4\) ......................................................... 39
3.9 Gadget for path factor problems ........................................... 40
3.10 A \(G\)-Factor including the connectors .................................. 41
3.11 A \(G\)-Factor excluding the connectors ................................. 41
3.12 Examples of augmenting configurations for the \({\bar{P}_1,}\bar{P}_2\}\)-packing problem ........................................ 45
3.13 Augmenting configuration components for the \({\bar{P}_1,}\bar{P}_2\}\)-packing problem ........................................ 46
3.14 Examples of augmented configurations for the \({\bar{P}_1,}\bar{P}_2\}\)-packing problem ........................................ 47
3.15 Example diagrams for case 2b .......................................... 53
3.16 Example diagrams for case 2c .......................................... 54
3.17 Example diagrams for case 3a .......................................... 57
3.18 Example diagrams for case 3b .......................................... 58
3.19 Gadget for the \(C_3\)-factor problem .................................. 60
3.20 A \(C_3\)-factor containing the connector vertices .................. 60
3.21 A \(C_3\)-factor excluding the connector vertices .................. 61
3.22 Gadget for the \(C_4\)-factor problem .................................. 62
3.23 A \(C_4\)-factor containing the connector vertices .................. 63
3.24 A \(C_4\)-factor excluding the connector vertices .................. 64
3.25 Gadget for the \(S_2\)-factor problem .................................. 64
3.26 Gadget for the \(S_3\)-factor problem .................................. 65
3.27 An \(S_3\)-factor containing the connector vertices ......... 66
3.28 An \(S_3\)-factor excluding the connector vertices .......... 66
3.29 Gadget for the \(S_4\)-factor problem .................................. 67
Chapter 1

Introduction

The aim of this thesis is two-fold. We will survey results on the general graph packing problem for undirected graphs and then we will begin to develop a similar theory for directed graphs.

1.1 Definitions and Notation

In order to understand the terminology of the graph packing problem, we must first introduce some definitions from graph theory and from matching theory.

A simple graph, or just graph, $H$ is an ordered pair $(V(H), E(H))$, where $V(H) = \{v_1, \ldots, v_n\}$ and $E(H) = \{e_1, \ldots, e_m\}$. The elements $v_i$, $i = 1 \ldots n$, of $V(H)$ are called vertices and the elements $e_i$, $i = 1 \ldots m$, of $E(H)$ are called edges. Each edge $e_i$ is an unordered pair of vertices. These two vertices are incident with the edge and adjacent to each other. The degree of a vertex is the number of edges with which it is incident. The size of the vertex-set of a graph, $|V(H)|$, is called the order of the graph.

Example: $H = (V(H), E(H))$

$V(H) = \{v_1, v_2, v_3, v_4, v_5\}$

$E(H) = \{e_1, e_2, e_3, e_4, e_5, e_6\}$

$e_1 = v_1v_3$, $e_2 = v_1v_4$, $e_3 = v_2v_3$

$e_4 = v_2v_5$, $e_5 = v_3v_4$, $e_6 = v_4v_5$

A visual representation of this graph is given in Figure 1.1.

A directed graph or digraph, $D$, is defined by the ordered pair $(V(D), A(D))$. Here,
V(D), is the set of vertices of D, and A(D) is the set of arcs or directed edges of D. Each arc is an ordered pair of vertices. Thus if \( e_i = v_jv_k \) is an arc, then the directed edge \( e_i \) begins in vertex \( v_j \) and ends in vertex \( v_k \).

A complete graph, or clique, on n vertices, \( K_n \), is a graph with vertex-set \{v_1, \ldots, v_n\} and edge-set \{v_iv_j|i \neq j\}. A complete bipartite graph, \( K_{m,n} \), is a graph with vertex-set \{v_1, \ldots, v_m\} \cup \{u_1, \ldots, u_n\} and edge-set \{v_iu_j|i \in \{1, \ldots, m\}, j \in \{1, \ldots, n\}\}. The complete bipartite graph \( K_{1,n} \) is called a star on n edges and is denoted \( S_n \). A path of length k, denoted \( P_k \), is a graph with vertex-set \{v_0, v_1, \ldots, v_k\} and edge-set \{e_1, \ldots, e_k\} such that \( e_i = v_{i-1}v_i \). A \((u,v)\)-path is a path such that \( v_0 = u \) and \( v_k = v \). Finally, a cycle of length k, denoted \( C_k \), is a graph with vertex-set \{v_1, \ldots, v_k\} and edge-set \{e_1, \ldots, e_k\} such that \( e_i = v_iv_{i+1}, i = 1, \ldots, k - 1 \), and \( e_k = v_kv_1 \). An example of each of these graphs is shown in Figure 1.2.

We will now introduce some necessary definitions from matching theory. Consider a graph \( H = (V, E) \). A graph \( K = (V', E') \) such that \( V' \subseteq V \) and \( E' \subseteq E \) is called a subgraph of \( H \) and we write \( K \subseteq H \). A matching \( M \) of a graph \( H \) is a set of pairwise vertex disjoint
edges of $H$. A vertex of $H$ which is incident with an edge of $M$ is covered by $M$ and is said to be in the matching $M$. The other vertices are called uncovered by $M$, or exposed in $M$. A path in $H$ is a subgraph $K$ of $H$ such that $K$ is a path. An alternating path in $H$ with respect to a matching $M$ in $H$, is a path in $H$ whose edges are alternately in and out of the matching $M$. An augmenting path is an alternating path which begins and ends in exposed vertices. The size of a matching is the number of vertices covered by the matching. A matching of largest size is called a maximum matching. If a matching covers all vertices of $H$, then it is a perfect matching or 1-factor of $H$. A perfect matching of $C_6$ is shown in Figure 1.3 where the bold edges represent edges in the matching.

![Figure 1.3: A perfect matching of $C_6$](image)

In matching theory, we usually search for maximum matchings or 1-factors of graphs. A result established by Berge [8] says that a matching $M$ of a graph $H$ is maximum if and only if $H$ contains no augmenting path with respect to $M$. Edmonds uses the search for augmenting paths approach to find an algorithm for maximum matchings in general graphs [20]. A theorem of Hall [26] for bipartite graphs gives a necessary and sufficient condition for the existence of a matching covering each vertex in one of the parts. Let $H$ be a bipartite graph with bipartition $(X,Y)$ and let $N(S)$ denote the neighbourhood of the set $S \subseteq X$. This neighbourhood is the set of all vertices of $H$ which are adjacent to vertices of $S$. Hall's theorem states that the bipartite graph $H$ contains a matching which covers all vertices of $X$ if and only if $|N(S)| \geq |S|$ for all $S \subseteq X$.

A necessary and sufficient condition for the existence of a perfect matching of an arbitrary graph was found by Tutte [45]. Before we give this condition, we introduce some necessary definitions. A non-empty graph $H$ is connected if for all $u,v \in V$ there exists a $(u,v)$-path. An induced subgraph $S$ of $H$ is a subgraph of $H$ with vertex set $V(S) \subseteq V(H)$ and edge set $E(S) \subseteq E(H)$ such that $E(S)$ is made up of those edges of $E(H)$ incident only with vertices in the vertex set $V(S)$. A subgraph $S$ of $H$ is a component of $H$ if $S$ is connected and there
does not exist an edge $uv$ where $u \in V(S)$ and $v \in V(H) \setminus V(S)$. A component is odd if the number of vertices in the component is odd. Now consider the graph $H \setminus S$ obtained from $H$ by removing all vertices of the set $S \subseteq V(H)$ and all edges incident with those vertices. Tutte proved that a graph $H$ has a perfect matching if and only if the number of odd components of $H \setminus S$ is less than or equal to $|S|$ for all $S \subseteq V(H)$.

The richness of these results from matching theory suggests that expanding the theory may also be fruitful. Let $\mathcal{G} = \{G_1, \ldots, G_k\}$ be a family of graphs. A $\mathcal{G}$-packing of a graph $H$ is a set of pairwise vertex-disjoint subgraphs of $H$ each of which is isomorphic to some graph in the family $\mathcal{G}$. A vertex of $H$ is covered by the $\mathcal{G}$-packing if it belongs to a subgraph of the $\mathcal{G}$-packing. If $\mathcal{G} = \{G\}$, then we call a $\mathcal{G}$-packing a $G$-packing. Since there is a natural correspondence between the edges of a graph and the set of subgraphs isomorphic to $K_2$, it is clear that every $K_2$-packing of a graph $H$ corresponds to a matching of $H$ and vice versa. The size of a packing is the number of vertices of $H$ covered by the packing. A $\mathcal{G}$-packing of $H$ of largest size is called a maximum $\mathcal{G}$-packing of $H$. If the $\mathcal{G}$-packing covers all vertices of $H$, then it is a perfect $\mathcal{G}$-packing or a $\mathcal{G}$-factor of $H$.

1.2 Background

The founders of matching theory are generally agreed to be Julius Petersen (1839–1910) and Dénes König (1884–1944). Petersen is best known for his results on regular graphs and König for his results on bipartite graphs. Other mathematicians instrumental in matching theory were Euler, Kirchhoff and Tait. A good history of the work of Petersen, König and their contemporaries can be found in the preface of Lovász and Plummer’s book Matching Theory [42]. An history of graph theory from 1736 to 1936 can be found in [10], and a “Sampler of major events of matching theory” can be found in [43].

Matching theory has many interesting applications in areas such as scheduling problems [27], assignment problems [37, 33], transportation problems and network flow problems [23, 33], the shortest path problem [33], the travelling salesman problem [13, 25], and the Chinese Postman problem [21]. Results from packing theory have useful applications to code optimization [11], clustering [32], and component placing [34]. Hell and Kirkpatrick considered one such application in [36] where they write that their interest in the study of generalized matchings stemmed from the study of scheduling examination periods.
1.3 The Problem

In this thesis, we study $G$-packing problems and $G$-factor problems. We have stated that the theory of packings is an extension of matching theory. The *maximum matching problem* is the problem of searching for a matching $M$ of a given input graph $H$, such that $M$ covers a maximum number of vertices. The analogous problem in packing theory, the *maximum $G$-packing problem*, which we will refer to simply as the $G$-packing problem, is the problem of searching for a maximum $G$-packing of an input graph $H$. Many polynomial algorithms exist for solution of the maximum matching problem, [20, 19, 22, 42], and it will be shown in Chapter 2 that many polynomial algorithms have also been found for the $G$-packing problem for various families $G$.

In matching theory, we study a special case of the maximum matching problem called the *perfect matching problem*. This is the problem of finding a perfect matching of the input graph $H$. Formally, a decision problem is a function from the instances of a problem to the answers "yes" and "no". For example, the perfect matching problem is formally defined as a decision problem below:

**Perfect Matching:**

*Instance:* A graph $H$

*Question:* Does $H$ admit a perfect matching, that is, is there a matching of $H$ of size $|V(H)|$?

The packing problem analogous to the perfect matching problem is the *$G$-factor problem*. It can be stated formally as a decision problem as follows:

**$G$-factor:**

*Instance:* A graph $H$

*Question:* Does $H$ admit a $G$-factor, that is, is there a packing of $H$ using graphs from the family $G$ such that the size of the packing is $|V(H)|$?

In Chapter 2, we survey results for the $G$-packing problem. In Chapter 3, we will study the $G$-factor problem in the directed case. This study is from the point of view of complexity theory, which is introduced in the following section.
1.4 Complexity Theory

To begin, we will define three broad complexity classes: the class \( P \); the class \( \mathcal{NP} \); the class \( \text{co-\mathcal{NP}} \). Complexity theory is a fundamental theory of computing science and as such a full introduction is beyond the scope of this thesis. This section serves as an introduction to the areas of complexity theory needed for this work. For a full introduction to complexity theory, the reader is referred to [24].

A problem is said to be in the class \( P \) if it can be solved by a polynomial time deterministic algorithm. Polynomial time means the number of steps executed by the algorithm is bounded by a polynomial in the size of the input. For example, a reasonable measure for the size of an input graph might be the number of vertices or the number of edges. Deterministic means that the \( i + 1 \)st step of the algorithm is uniquely determined by the first \( i \) steps.

The class \( \mathcal{NP} \) contains decision problems which can be solved in polynomial time by a nondeterministic algorithm. A nondeterministic algorithm has two stages: the guessing stage and the checking stage. There are two possible outcomes of a nondeterministic algorithm. Either the instance of the decision problem is a yes-instance (the answer to the question is yes), in which case the checking stage of the algorithm stops and answers yes, or else the instance is not a yes-instance, in which case, the algorithm may stop and answer no, or it may never stop. For example, consider a nondeterministic algorithm for the perfect matching problem. Let the input graph \( H \) have \(|V(H)| = n\) and let \( i = 1, \ldots, n/2 \). In the guessing stage, choose an edge of \( H \) at each of the \( i \) steps of the algorithm. This gives \( \frac{n}{2} \) edges and this guess at a solution is called a certificate. The certificate must be polynomial in size. In the checking stage, check whether or not these \( \frac{n}{2} \) edges form a perfect matching. This part of the algorithm must complete in time polynomial in the size of the input. In our example, determining whether our certificate is a perfect matching of \( H \) can be achieved in polynomial time. A yes-certificate of a problem is a certificate which upon testing gives the answer yes to the decision problem question. In our example, a yes-certificate is a perfect matching. An instance of the problem is called a yes-instance if there exists a yes-certificate.

Any decision problem solvable by a polynomial time deterministic algorithm is solvable by a polynomial time nondeterministic algorithm since we can simply use the deterministic algorithm as the checking stage of the nondeterministic algorithm with an empty guess. Therefore, the problems in the class \( P \) are also in the class \( \mathcal{NP} \) and so \( P \subseteq \mathcal{NP} \).
CHAPTER 1. INTRODUCTION

The class \( co - \mathcal{NP} \) contains the complement problems, \( \Pi^c \), to those problems \( \Pi \) in the class \( \mathcal{NP} \). Formally, if \( \Pi \) denotes a decision problem, then \( co - \mathcal{NP} = \{ \Pi^c | \Pi \in \mathcal{NP} \} \). The problem \( \Pi^c \) is the same problem as \( \Pi \) but with the answers reversed. As explained in [24], "If the problem 'Given \( I \), is \( X \) true for \( I \)?' can be solved by a polynomial time deterministic algorithm, then so can the complementary problem 'Given \( I \), is \( X \) false for \( I \)?' This is because a deterministic algorithm halts for all inputs, so all we need to do is interchange the 'yes' and 'no' responses". A polynomial transformation from one decision problem \( A \) to another decision problem \( B \) is a function \( f \) from the instances \( a \) of \( A \) to the instances \( b \) of \( B \) such that this function can be computed in polynomial time and for all instances of \( A \), the instance is a yes-instance if and only if \( f(a) \), is a yes-instance of \( B \). It is not clear that all problems can be polynomially transformed to their complement problems. Many problems in \( co - \mathcal{NP} \) do not seem to be in \( \mathcal{NP} \) and so it is conjectured that \( \mathcal{NP} \neq co - \mathcal{NP} \). It has also been conjectured that \( \mathcal{P} = \mathcal{NP} \cap co - \mathcal{NP} \), but this conjecture remains an open problem.

In this thesis, we will prove \( \mathcal{NP} \)-completeness of various packing problems. In order to define what is meant by \( \mathcal{NP} \)-complete, we must introduce the concept of reducing one problem to another. Consider the following decision problems \( A \) and \( B \):

\[
A: \quad \text{Instance: } a \\
\text{Question: Is } a \text{ this way?}
\]

\[
B: \quad \text{Instance: } b \\
\text{Question: Is } b \text{ that way?}
\]

We will be using the term polynomial time reduction, or simply reduction in this thesis. This technique is defined in [24] as follows: "The principal technique used for demonstrating that two problems are related is that of "reducing" one to the other, by giving a constructive transformation that maps any instance of the first problem into an equivalent instance of the second. Such a transformation provides the means for converting any algorithm that solves the second problem into a corresponding algorithm for solving the first problem." Therefore, if we have a polynomial time reduction from problem \( A \) to problem \( B \), then a polynomial-time algorithm for \( B \) implies that there is a polynomial-time algorithm for \( A \). We denote a polynomial time reduction of problem \( A \) to problem \( B \) by \( A \preceq B \).

Now we can define the complexity class of \( \mathcal{NP} \)-complete problems. A problem is \( \mathcal{NP} \)-complete if it is in the class \( \mathcal{NP} \) and if all other problems in the class \( \mathcal{NP} \) can be polynomially reduced to it. The foundations of \( \mathcal{NP} \)-completeness theory were laid in [14]. Stephen Cook emphasized the importance of polynomial time reductions. He gave the first proof that the
satisfiability problem (SAT) had the property that every other problem in the class \( \mathcal{NP} \) could be polynomially reduced to it. This property was found to be true of a number of problems and became known as \( \mathcal{NP} \)-complete. The class of \( \mathcal{NP} \)-complete problems is a subset of \( \mathcal{NP} \) containing the hardest problems of the class \( \mathcal{NP} \). Considering Cook's result, in order to prove \( \mathcal{NP} \)-completeness, we do not have to reduce every problem in \( \mathcal{NP} \) to our problem. Instead, we use the following theorem.

**Theorem 1.4.1.** [24] Let \( \Pi_1 \) and \( \Pi_2 \) be problems. If \( \Pi_1 \preceq \Pi_2 \) and \( \Pi_2 \in \mathcal{NP} \), then \( \Pi_1 \) is \( \mathcal{NP} \)-complete implies \( \Pi_2 \) is \( \mathcal{NP} \)-complete.

Thus, to prove that a problem \( \Pi_2 \) is \( \mathcal{NP} \)-complete, we first show that it is in \( \mathcal{NP} \) and then show that a known \( \mathcal{NP} \)-complete problem \( \Pi_1 \) can be polynomially reduced to the problem \( \Pi_2 \). There are various ways to do this reduction in order to prove \( \mathcal{NP} \)-completeness. The three techniques which are most frequently used are *local replacement*, *restriction*, and *component design* [24]. In this thesis, we will use the local replacement technique.

Finally, a problem \( \Pi \) is in the class \( \mathcal{NP} \)-hard if some \( \mathcal{NP} \)-complete problem can be polynomially reduced to it.
Chapter 2

Survey of the General Graph Packing Problem

The $G$-packing problem has been extensively studied over the last few decades. In this survey, we will consider results on the general undirected graph packing problem which asks “How many vertices of $H$ can we cover with copies of graphs in the family $G$?” In addition to the general graph packing problem, many other packing problems have been defined and studied.

In this chapter, we will separate results according to the type of family $G$ considered. We begin by considering families of complete graphs and then move on to families of complete bipartite graphs, of paths, of cycles, and finally to mixed families. The results in this chapter can be found in \cite{4, 5, 15, 17, 24, 27, 28, 29, 30, 36, 38, 39, 40}.

2.1 Families of Complete Graphs

Consider the graph packing problem for $G = \{K_1\}$. This is clearly polynomial since for any graph, the set of vertices is a perfect $\{K_1\}$-packing. The $G$-packing problem for $G = \{K_2\}$ is also polynomial since this is simply the maximum matching problem.

The next family to consider is $G = \{K_3\}$. This is sometimes referred to as the triangle packing problem and it is $\mathcal{NP}$-complete \cite{17}. The $\mathcal{NP}$-completeness proof can be done by reduction of the 3-dimensional matching problem (3DM) to the $\{K_3\}$-factor problem as shown below. The problem 3DM is an $\mathcal{NP}$-complete decision problem which looks for a
3-dimensional matching of a set of triples $\mathcal{T}$. This problem can be stated as follows:

**3DM:**
- **Instance:** An integer $q$, sets $W$, $X$, and $Y$ with $|W| = |X| = |Y| = q$, and a subset $\mathcal{T}$ of $W \times X \times Y$.
- **Question:** Is there a subset $M$ of $\mathcal{T}$ containing $q$ triples no two of which use the same element from any of the sets $W$, $X$ or $Y$?

The next result is stated in [29] and we give our own proof below.

**Theorem 2.1.1.** The $\{K_3\}$-factor problem is $NP$-complete.

**Proof:** The $\{K_3\}$-factor problem is in $NP$ since given a potential $\{K_3\}$-factor of a graph $H$ as a certificate, we can verify in polynomial time that all vertices of $H$ are covered by exactly one $K_3$. We will reduce the $NP$-complete problem $3DM$ to the $\{K_3\}$-factor problem to establish $NP$-completeness.

Let $\mathcal{T} = \{(t_1, t_2, t_3) | t_1 \in W, t_2 \in X, t_3 \in Y\}$ be some subset of $W \times X \times Y$ where $W, X, Y = \{1, 2, \ldots, q\}$. We shall construct a graph $G$ such that the set of triples $\mathcal{T}$ has a 3DM if and only if $G$ admits a $\{K_3\}$-factor.

In $NP$-completeness proofs we will often use the local replacement technique. In this technique, we use gadgets to construct an instance of our problem. In this proof we will use the gadget shown in Figure 2.1. This gadget is used to construct a graph $G$.

![Gadget for the $\{K_3\}$-factor problem](image)

**Figure 2.1:** Gadget for the $\{K_3\}$-factor problem

**Construction:** Consider a graph having vertex set $W \cup X \cup Y$. We construct
the edge set as follows. For each triple \((t_1, t_2, t_3)\) of \(\mathcal{T}\), identify the connector vertices, \(a\), \(b\), and \(c\), of a copy of the gadget with the vertices \(t_1\), \(t_2\) and \(t_3\), respectively, of the graph. All other vertices, the interior vertices, of each copy of the gadget become vertices and the edges of the copies of the gadget become edges of the graph. Call the resulting graph \(G\).

**Claim 2.1.1.** The set \(\mathcal{T}\) has a three dimensional matching if and only if \(G\) has a \(\{K_3\}\)-factor.

*Proof:* Suppose \(\mathcal{T}\) has a 3DM. Then for triples \((t_1, t_2, t_3)\) in the matching, the gadget with connector vertices \(a = t_1\), \(b = t_2\) and \(c = t_3\) has the \(\{K_3\}\)-factor shown in Figure 2.2.

![Figure 2.2: A \(\{K_3\}\)-factor containing the connector vertices](image)

For triples not in the matching, that is when \(t_1\), \(t_2\), and \(t_3\) are each covered by other triples, the copy of the gadget having vertices \(t_1\), \(t_2\) and \(t_3\) has the \(\{K_3\}\)-factor of Figure 2.3.

Thus, there is always a \(\{K_3\}\)-factor of \(G\) when there is a 3DM of \(\mathcal{T}\).

Now, suppose \(G\) has a \(\{K_3\}\)-factor. Consider some copy of the gadget. The point \(x_3\) must either be covered by the triangle \(x_1x_2x_3\) or else by the triangle \(y_3z_3x_3\). Suppose \(x_3\) is covered by \(y_3z_3x_3\). Then in order to cover the vertex \(x_1\), we must choose the triangle \(ax_1x_2\). Similarly, we must choose \(cz_1z_2\) and \(by_1y_2\) and we have a \(\{K_3\}\)-factor of the gadget as in Figure 2.2. In this case, since the connector vertices are covered
by triangles within the gadget, we choose the triple \((a, b, c)\) to be in the 3-dimensional matching of \(\mathcal{T}\).

Now, suppose that \(x_3\) is covered by the triangle \(x_1x_2x_3\). Then vertex \(y_3\) must be covered by the triangle \(y_1y_2y_3\) and the vertex \(z_3\) by the triangle \(z_1z_2z_3\). In this case, the connectors are not covered by triangles in the gadget and so we do not choose \((a, b, c)\) as a triple of the 3-dimensional matching.

The resulting set of chosen triples \(\mathcal{T} = \{(a, b, c)\}\) is a 3-dimensional matching. □

Since 3DM is an \(\mathcal{NP}\)-complete problem and we have shown that it reduces to the \(\{K_3\}\)-factor problem, the \(\{K_3\}\)-factor problem is also \(\mathcal{NP}\)-complete. □

Hell and Kirkpatrick [35, 36] have proven that the problem of packing an input graph with a connected graph other than \(K_1\) or \(K_2\) is \(\mathcal{NP}\)-complete. However, when we consider packing with a graph \(G\) which is the disjoint union of \(\alpha\) copies of \(K_1\) and \(\beta\) copies of \(K_2\) then the \(G\)-packing problem is again polynomial. Consider the following theorem:

**Theorem 2.1.2.** [36] Let \(G\) be the disjoint union of \(\alpha\) copies of \(K_1\) and \(\beta\) copies of \(K_2\). Then \(H\) admits a \(G\)-factor if and only if \(|V(G)|\) divides \(|V(H)|\) and some matching of \(H\) has at least \(\beta \frac{|V(H)|}{|V(G)|}\) edges.

**Proof:** Suppose \(H\) admits a \(G\)-factor. If \(|V(G)|\) does not divide \(|V(H)|\) then we cannot cover all vertices of \(H\) with copies of \(G\) such that any vertex of \(H\) is
in exactly one copy of G. Therefore, |V(G)| divides |V(H)|. Within the set of
vertices of H used by each copy of G of the packing, there needs to be \( \beta \) copies
of \( K_2 \) since G has this number of \( K_2 \)'s. This is true for each of the disjoint copies
of G, of which there are \( \frac{|V(H)|}{|V(G)|} \), and so in total, the number of edges of some
matching of H must have at least \( \beta \frac{|V(H)|}{|V(G)|} \) edges. Conversely, suppose |V(G)|
divides |V(H)| and in addition, some matching \( M \) of H has at least \( \beta \frac{|V(H)|}{|V(G)|} \)
edges. Then we cover vertices of H in the following way: choose \( \beta \) edges from
\( M \) and some \( \alpha \) vertices from \( H \setminus M \) and let this be a copy of G in the G-packing.
These chosen edges and vertices cannot be used again. Since there are \( \beta \frac{|V(H)|}{|V(G)|} \)
edges in the matching, we can define \( \frac{|V(H)|}{|V(G)|} \) copies of G in this way. Clearly, the
G-packing is perfect and so this defines a G-factor of H. \( \square \)

For graphs G satisfying the hypothesis of Theorem 2.1.2, the algorithms used to find a
maximum matching can be used to determine if H admits a G-factor. Since determining
the hypotheses of Theorem 2.1.2 can be done in polynomial time, we have the following
corollary:

**Corollary 2.1.1.** [36] If \( G \) is the disjoint union of \( \alpha \) copies of \( K_1 \) and \( \beta \) copies of \( K_2 \), then
the G-factor problem is polynomial.

It is also proven in [36] that for all other graphs G, the G-factor problem is \( \mathcal{NP} \)-complete.

\( \{K_2, K_3\} \)-Packings

The packing problem with \( G = \{K_2, K_3\} \) has been studied quite extensively, see [17, 29],
and has been found to be polynomial. Hell and Kirkpatrick treat the \( \{K_2, K_3\} \)-packing
problem in [29] and prove the following theorem.

**Theorem 2.1.3.** [29] Let \( G = \{K_2, K_3\} \). Then there exists a polynomial-time algorithm to
find a maximum size G-packing of arbitrary input graph H.

In the proof of Theorem 2.1.3, the authors use Lemma 2.1.1 below which is analogous
to the result from matching theory that a matching of H is maximum if and only if H has
no augmenting path. Before we state the lemma, some more definitions are required. First,
consider G a \( \{K_2, K_3\} \)-packing of a graph H. Edges of H which are in the packing are
called G-edges and subgraphs of H in the packing which are isomorphic to \( K_3 \) are called
An alternating path of $H$ with respect to $G$ is a path in $H$ whose edges are alternately in $G$ and not in $G$. That is, if the first edge of the path is in the packing, then the next edge of the path is not in the packing, et cetera. An augmenting path of $H$ with respect to $G$ is an alternating path of $H$ with respect to $G$ between two exposed vertices of $H$. An augmenting tail of $H$ with respect to $G$ is an alternating path of $H$ with respect to $G$ between a vertex $v_0$ which is exposed in $G$ and a vertex $v_{n+1}$ which is covered in $G$, such that $v_{n+1}$ is covered by the $G$-triangle $v_{n+1}v_{n+2}v_{n+3}$. Finally, an augmenting kite of $H$ with respect to $G$ is an even length alternating path of $H$ with respect to $G$ between an uncovered vertex $v_0$ and some vertex $v_n$ covered by the $G$-edge $v_{n-1}v_n$, together with a $K_3$ $v_nv_{n+1}v_{n+k+1}$, and an alternating path $v_{n+1}, \ldots, v_{n+k+1}$. A degenerate kite is a kite with $n = 0$ and $k = 1$. That is, a triangle with one edge covered by the packing. These augmenting configurations are shown in Figure 2.4. Note that circled vertices denote vertices which are exposed with respect to the packing.

Augmentation of these configurations is done in the following manner. For an augmenting path, we use a switching technique in which all $G$-edges are removed from $G$ and non $G$-edges are made $G$-edges. This switching technique is also used to augment the tail along the path $v_0, \ldots, v_{n+1}$. In addition, in the case of the tail, edges $v_{n+1}v_{n+2}$ and $v_{n+1}v_{n+3}$ become non-packing edges, but edge $v_{n+2}v_{n+3}$ remains a packing edge. The kite is augmented by switching along the paths $v_0, \ldots, v_n$, and $v_{n+1}, \ldots, v_{n+k+1}$, and by adding the

---

**Figure 2.4:** Augmenting configurations for the \{K_2, K_3\}-packing problem
Lemma 2.1.1. [29] A \(K_2, K_3\)-packing \(G\) of a graph \(H\) is of maximum size if and only if it admits no augmenting path, tail or kite.

Proof: Clearly, if \(G\) is of maximum size then there can be no augmenting configurations.

Suppose \(H\) is a smallest graph which admits a packing \(G\) which has no augmenting configurations and another packing \(G'\) of \(H\) which covers more vertices than \(G\). Then there must be a vertex \(v\) which is covered by \(G'\) but not by \(G\). We will consider two cases:

1. The vertex \(v\) is covered by a \(G'\)-edge \(vw\).

Since \(G\) has no augmenting configuration, \(v\) cannot be on an edge with a vertex which is exposed in \(G\), and so \(w\) must be covered by \(G\). Since there are no augmenting configurations, \(w\) cannot be covered by a \(G\)-triangle and so must be covered by a \(G\)-edge \(wu\).

Consider the graph \(\tilde{H}\) with \(V(\tilde{H}) = V(H) \setminus \{v, w\}\) and \(E(\tilde{H}) = E(H) \setminus \{\text{edges incident with } v \text{ or } w\}\), and the packings \(G\) and \(G'\) restricted to \(\tilde{H}\).

In \(\tilde{H}\), \(G\) covers two less vertices, namely \(w\) and \(u\), and \(G'\) covers two less vertices, namely \(v\) and \(w\). Therefore, \(G'\) covers more vertices of \(\tilde{H}\) than \(G\) and so by the choice of \(H\), \(G\) must admit an augmenting configuration in \(\tilde{H}\). If this augmenting configuration begins at a vertex different from \(u\) and does not involve \(u\), then it would be an augmenting configuration of \(G\) in \(H\) as well, but this contradicts the definition of \(G\). If the augmenting configuration begins at a vertex other than \(u\) and ends at \(u\), then it is an augmenting configuration ending at \(v\) in \(H\). If the augmenting configuration begins at \(u\), then in \(H\) we would have an augmenting configuration of \(G\) beginning at \(v\) with the edge \(vw\) and followed by the \(G\)-edge \(wu\). Again this contradicts the definition of \(G\).

2. The vertex \(v\) is covered by a \(G'\)-triangle \(vwz\). Since \(G\) has no augmenting path or tail, we cannot have any of the graphs of Figure 2.5.
Therefore, both \( w \) and \( z \) must be covered by \( G \)-edges. However, \( w \) and \( z \) cannot be incident with the same \( G \)-edge since \( G \) has no degenerate kites. Thus, there are two distinct \( G \)-edges as in Figure 2.6.

Consider the graph \( \tilde{H} \) with \( V(\tilde{H}) = V(H) \setminus \{v, w, z\} \), and \( E(\tilde{H}) = E(H) \setminus \{ \text{edges incident with } v, w, \text{ or } z \} \), and the packings \( G \) and \( G' \) restricted to \( \tilde{H} \). In \( \tilde{H} \), \( G \) covers four less vertices than in \( H \) and \( G' \) covers three less vertices. Therefore, \( G' \) covers more vertices of \( \tilde{H} \) than \( G \). So, by the choice of \( H \), \( G \) must have an augmenting configuration in \( \tilde{H} \). If the augmenting configuration is a path between \( x \) and \( y \), then in \( H \) we would have the augmenting configuration of \( G \) shown in Figure 2.7, which is an augmenting kite in \( H \). This is a contradiction with the definition of \( G \).

If the augmenting configuration begins at \( x \) (or \( y \)), then it is an augmenting configuration of \( G \) beginning at \( v \) in \( H \), which again contradicts the choice of \( G \). If the configuration begins at some point other than \( x \) or \( y \), then it is an augmenting configuration of \( G \) in \( H \) as well, another contradiction.

Therefore, the packing \( G \) must be of maximum size. \( \Box \)

The augmenting configurations of Lemma 2.2.1 can be used in an algorithm to find
a maximum size packing in polynomial time. Before we give the algorithm, we need the following proposition which uses some new terminology. A set of vertex disjoint triangles of $H$ will be denoted $T$.

Suppose that a given packing $G$ is not of maximum size. Then it follows from Lemma 2.1.1 that $G$ admits an augmenting path, tail or kite and therefore there is a larger packing $G'$ whose set of triangles differs by at most one triangle from the set of triangles of $G$. This is written formally in the following theorem.

**Proposition 2.1.1.** [29] Let $G$ be a $\{K_2, K_3\}$-packing of $H$, $T$ the set of $G$-triangles. Then $G$ is a $\{K_2, K_3\}$-packing of maximum size if and only if no $\{K_2, K_3\}$-packing $G'$ of $H$ covers more vertices than $G$ where the set of $G'$-triangles:

1. is equal to $T$;
2. is equal to $T \setminus \{uvw\}$ for some triangle $\{uvw\} \in T$;
3. is equal to $T \cup \{uvw\}$ for some triangle $\{uvw\} \notin T$.

**Proof:** If $G$ is of maximum size, then clearly there is no packing $G'$ covering more vertices than $G$.

Suppose that $G$ is not of maximum size. Then by Lemma 2.1.1 there must be an augmenting configuration resulting in a larger packing $G'$. If the configuration were a path, then $G'$ would have the same number of triangles as $G$ (case 1). If the configuration were a tail, then $G'$ would have one less triangle than $G$ (case 2). Finally, if the configuration were a kite, then $G'$ would have one more triangle than $G$ (case 3). □

Proposition 2.1.1 and Lemma 2.1.1 lead to a polynomial-time algorithm for finding a maximum $\{K_2, K_3\}$-packing. In this algorithm, $G$ denotes a $\{K_2, K_3\}$-packing of $H$, and $T$ denotes the set of $G$-triangles.

![Figure 2.7: Augmenting path between $x$ and $y$](image-url)
Algorithm 2.1.1. To find a maximum size \(\{K_2, K_3\}\)-packing of a graph \(H\).

\[
\begin{align*}
\text{begin} & \\
G & := \text{maximum matching of } H \\
T & := \emptyset \\
\text{while (maximum size of a } \{K_2, K_3\}\text{-packing } G' \text{ with set } T' \text{ of } G'\text{-triangles, where } T' = T \setminus \{uvw\} \text{ for some } \{uvw\} \in T, \text{ or } T' = T \cup \{uvw\} \text{ for some } \{uvw\} \notin T > \text{size of } G \text{ do} & \\
G & := G' \\
T & := T' \\
\text{return } G & \\
\text{end}
\end{align*}
\]

In this algorithm, the set of triangles, \(T'\) of the packing \(G'\) is related to the set of triangles \(T\) of \(G\) as follows: \(T' = T \setminus \{uvw\}\), where \(uvw \in T\), or \(T' = T \cup \{uvw\}\), where \(uvw \notin T\). The set \(T'\) and consequently the packing \(G'\), can be found in one of two ways. Firstly, we systematically choose one triangle \(uvw\) from \(T\) and remove it from \(T\) to obtain \(T^-\). We then run a maximum matching algorithm on \(H \setminus T^-\). The resulting matching together with \(T^-\) gives us a packing. If this packing covers more vertices than \(G\), then it is a packing \(G'\) and \(T^- = T'\). We then update \(G\) and \(T\) and continue. If not, we choose a different triangle of \(T\) to remove and proceed as before. If no larger packing \(G'\) is found, we move on to the triangles which are not in \(T\). This time, we systematically add a triangle \(uvw\) which is not in \(T\) to \(T\) to obtain \(T^+\). Then as before, we run a maximum matching algorithm on \(H \setminus T^+\) and the resulting matching along with \(T^+\) gives us a packing. If this packing is larger than \(G\), then it is a packing \(G'\) and \(T^+ = T'\). We then update \(G\) and \(T\) and run the while loop again. If no packing \(G'\) is found which is larger than \(G\), then \(G\) is a maximum packing and we are done. This process is polynomial, since if \(n = V(H)\), the number of triangles in \(T\) at any time is at most order \(n\), as no vertices can be repeated, the number of triangles not in \(T\) at any time is at most order \(n^3\), and finding a maximum matching of the remaining vertices can be found in polynomial time \([20]\).

### 2.1.1 \(\{K_2, F_i\}\)-Packings

Proposition 2.1.1, Lemma 2.1.1, and the algorithm can be extended to other cases including the case of Theorem 2.1.4 below where \(G = \{K_2, F_1, \ldots, F_k\}\) and all the \(F_i\) are hypomatchable. A graph \(H\) is hypomatchable if \(H\) has no perfect matching, but \(H \setminus v\) does have a
Theorem 2.1.4. [29] Let $G = \{K_2, F_1, \ldots, F_k\}$ and assume that each $F_i$ is hypomatchable. Then there exists a polynomial-time algorithm to find a maximum size $G$-packing of arbitrary input graph $H$.

This theorem will not be proven here. The proof given in [29] uses an extension of Lemma 2.1.1. In order to extend Lemma 2.1.1, we need to extend the augmenting configurations to the case $G = \{K_2, F_1, \ldots, F_k\}$. Let $G$ and $G'$ be $G$-packings and let an alternating path refer to a $G, G'$-alternating path in $H$ whose edges are alternately $G$-edges and $G'$-edges. The augmenting configurations are then as follows.

1. An augmenting path is an alternating path between two vertices exposed in $G$.

2. An augmenting tail is an alternating path between an exposed vertex of $G$ and a vertex covered in $G$ by some $G - F_i$ (not by a $G$-edge).

3. An augmenting kite is an even length alternating path between an exposed vertex $u$ of $G$ and a vertex $v$ of some $F_i$ such that the $F_i$ and the alternating path intersect only at $v$. Also, there exist $\frac{1}{2} |V(F_i)| - 1$ vertex-disjoint alternating paths between pairs of vertices of $F_i - v$ such that each path begins and ends with a $G$-edge. An example of such a kite is given in Figure 2.8.

In this Figure, the $\frac{1}{2} |V(F_i)| - 1$ vertex-disjoint alternating paths are the paths 1, 3, and 2, 6, 5, 4.

The following proposition is analogous to Lemma 2.1.1.

![Figure 2.8: The kite for $G = \{K_2, C_5\}$](image-url)
Proposition 2.1.2. [29] Let $G$ satisfy the assumptions of Theorem 2.1.4. The $G$-packing $G$ of a graph $H$ is of maximum size if and only if it has no augmenting path, tail, or kite in $H$.

There is a corollary to Theorem 2.1.4 which considers a possibly infinite family of cliques.

Corollary 2.1.2. [27, 29, 17, 36] If $G \subseteq \{K_1, K_2, K_3, \ldots\}$ ($G$ not necessarily finite) and $K_1 \in G$ or $K_2 \in G$, then there is a polynomial-time algorithm for the maximum size $G$-packing problem.

Proof: Clearly $K_1 \in G$ is trivial and $G = K_2$ is solvable by a polynomial maximum matching algorithm. Then consider the case where $K_2$ is in $G$ and all other cliques in $G$ are even. All elements of $G$ have perfect matchings and so a maximum size $G$-packing is found by searching for a maximum size $\{K_2\}$-packing.

If $K_2$ is in $G$ and $K_t, t > 1$, is the smallest odd clique in $G$, then any $G$-packing has a $\{K_2, K_t\}$-factor since odd cliques can be covered with a copy of $K_t$ and some number of copies of $K_2$, and any even clique can be covered with copies of $K_2$. Therefore, it is sufficient to find a maximum size $\{K_2, K_t\}$-packing which by Theorem 2.1.4 can be done in polynomial time. □

2.2 Families of Complete Bipartite Graphs

Recall that a complete bipartite graph having a partition of size one and the other of size $k$ is called a star and is denoted $S_k$. The sets $\{S_1, S_2, \ldots, S_k\}$ and $\{S_1, S_2, \ldots\}$ are called sequential star sets.

The search for a polynomial algorithm to find a maximum $\{S_1, \ldots, S_k\}$-packing stemmed from an observation of Las Vergnas, given below, concerning $(1, k)$-factors. A $(1, k)$-factor is a spanning subgraph with all degrees between 1 and $k$.

Proposition 2.2.1. [47] A graph has a $\{S_1, \ldots, S_k\}$-factor if and only if it has a $(1, k)$-factor.

Proof: Given a graph $H$ which has no isolated vertices, consider the following $O(|E|)$ algorithm.
Algorithm 2.2.1. To find a \{S_1, \ldots, S_k\}-factor or \{S_1, S_2, \ldots\}-factor of \( H \).

1. For each edge \( uv \in H \), delete \( uv \) if both \( u \) and \( v \) have degree greater than one.
2. Update the degrees of \( u \) and \( v \).

If all degrees of the vertices of \( H \) are between 1 and \( k \), then this algorithm will find an \{\( S_1, \ldots, S_k \)\}-factor. Thus a \((1, k)\)-factor found by any algorithm for finding \((1, k)\)-factors [22, 44], can be modified to give an \{\( S_1, \ldots, S_k \)\}-factor. Since the \((1, k)\)-factor problem is polynomial, the \{\( S_1, \ldots, S_k \)\}-factor problem is also polynomial.

Note that if the algorithm is applied to an arbitrary graph \( H \), then it will find a maximum sequential star packing, that is an \{\( S_1, S_2, \ldots \)\}-packing. Therefore, the \{\( S_1, S_2, \ldots \)\}-packing problem is polynomial.

It is proven in [28, 30] that for any other families \( G \) of complete bipartite graphs, the \( G \)-packing problem is \( \mathcal{NP} \)-complete or harder. We will not prove this here.

Proposition 2.2.2. [28, 30] A graph \( H \) has an \{\( S_1, S_2, \ldots \)\}-factor if and only if it has no isolated vertices.

Proof: This follows from Proposition 2.2.1. Since each vertex must have degree at least 1, no vertex can be isolated. \( \Box \)

The augmenting configurations referred to in Theorem 2.2.1 below are shown in Figure 2.9. The number or letter beside each vertex denotes the degree of that vertex in the packing. Brackets with an asterisk around a part of the configuration denote that that part of the configuration may be repeated \( q \) times, \( q \geq 0 \). As before, bold edges denote edges covered by the packing and circled vertices denote vertices uncovered in the packing.

When we augment these configurations, we obtain the configurations of Figure 2.10.

Theorem 2.2.1. [28, 30] Let \( k \geq 2 \). The size of a maximum \{\( S_1, \ldots, S_k \)\}-packing of \( H \) is equal to the minimum, over all \( T \subseteq V(H) \), of \( n + k \cdot |T| - i_T \) where \( n = |V(H)| \) and \( i_T \) denotes the number of isolated vertices of \( H \setminus T \).
Proof: If $T$ is a subset of $V(H)$, then the largest number of vertices in $H \setminus T$ which are not exposed in $H$ occurs if each vertex in $T$ is the centre vertex of an $S_k$ and all the other vertices of the $S_k$ are in $H \setminus T$. This number is $k - |T|$. Therefore, the number of exposed vertices in the packing $G$ of $H$ is at least the number of vertices which are exposed in $H$ and this number is $i_T - k \cdot |T|$. Therefore, the number of covered vertices in the packing $G$ is at most $n - (i_T - k \cdot |T|) = n + k \cdot |T| - i_T$ for any $T \subseteq V(H)$. Now, to prove this equality we define a packing $G$ and a set $T \subseteq V(H)$ such that $G$ covers exactly $n + k \cdot |T| - i_T$ vertices of $H$.

Let $G$ be any $\{S_1, \ldots, S_k\}$-packing of $H$ which admits no augmenting configuration. Let $U$ denote the set of exposed vertices in $G$. Let the sets $T$ and $W$ be defined as follows:

If $\deg_G v = k$ and $v$ is adjacent to a vertex of $U \cup W$ in $H$, then $v \in T$. 

Figure 2.9: Augmenting star configurations

Figure 2.10: Augmented star configurations
If $w$ is adjacent to a vertex of $T$ in $G$, then $w \in W$.

Since $G$ admits no augmenting configuration, each neighbour of a vertex $u \in U$ must have degree $k$ with respect to $G$ and so be an element on $T$. Also, any $w \in W$ must have degree 1 in $G$ and be reached from some $u$ by a path on which the degrees of the vertices alternate between $k$ and 1 with the vertex adjacent to $u$ having degree $k$ and the vertex preceding $w$ having degree $k$. Since the vertex preceding $w$ has degree $k$ and is adjacent to $w \in W$, it must be in $T$ and so each neighbour of a $w \in W$ is an element of $T$. Consequently, all vertices of $U \cup W$ must be isolated in $H \setminus T$, and so $i_T \geq |U| + |W|$. This means that $G$ covers $n - |U| > n + |W| - i_T = n + k \cdot |T| - i_T$ vertices. $\square$

**Corollary 2.2.1.** [28, 30] An $\{S_1, \ldots, S_k\}$-packing $G$ of $H$ is maximum if and only if it admits no augmenting configuration.

**Proof:** Clearly, if $G$ admits an augmenting configuration then $G$ is not maximum. If $G$ does not admit an augmenting configuration, then by the proof of Theorem 2.2.1, there exists a set $T \subseteq V(H)$ such that $G$ covers $n + k \cdot |T| - i_T$ vertices of $H$ and so is a maximum packing. $\square$

Another corollary to Theorem 2.2.1, given below, was proven independently by Amahashi and Kano [6] and by Las Vergnas [47]. It can also be found in [3, 9].

**Corollary 2.2.2.** [28, 30] Let $k \geq 2$. A graph $H$ admits an $\{S_1, \ldots, S_k\}$-factor if and only if it does not have a set $T$ of vertices whose deletion results in more than $k \cdot |T|$ isolated vertices.

**Proof:** [28, 30] By Theorem 2.2.1, the maximum size of a packing of $H$ is the minimum of $n + k \cdot |T| - i_T$ over all $T$. If there is a set $T$ such that $i_T > k \cdot |T|$, then $n + k \cdot |T| - i_T < n$. Therefore, the maximum size of a packing will be $n$ if and only if $i_T \leq k \cdot |T|$ for each $T \subseteq V(H)$. $\square$

### 2.3 Families of Paths

Having considered families of complete graphs and of complete bipartite graphs, we now shift our attention to families of paths. Recall that in this thesis, $P_k$ represents the path with $k$ edges.
The maximum $G$-packing problem where $G$ is the path on one edge, $P_1$, is equivalent to the maximum matching problem which is well known to be polynomial.

**The Family $\{P_2\}$**

It follows from Theorem 2.1.1 that the $\{P_2\}$-packing problem is $NP$-complete. We now give a proof of this using the local replacement technique.

**Theorem 2.3.1.** The $\{P_2\}$-factor problem is $NP$-complete.

**Proof:** This problem is in $NP$ since if we are given a potential $\{P_2\}$-factor of a graph $H$ as a certificate, we can verify that all vertices of $H$ are covered by exactly one $P_2$ in polynomial time. We will reduce the $NP$-complete problem 3DM to the $\{P_2\}$-factor problem to establish the $NP$-completeness. Consider the gadget of Figure 2.11.

![Figure 2.11: Gadget for the $\{P_2\}$-factor problem](image)

**Construction:** Recall that 3DM considers a set $T = \{(t_1, t_2, t_3) | t_1 \in W, t_2 \in X, t_3 \in Y\}$, some subset of $W \times X \times Y$ where $W, X, Y = \{1, 2, \ldots, q\}$. We will construct a graph $G$ as follows: Let $W \cup X \cup Y$ be the vertices of a graph. For each triple $(t_1, t_2, t_3)$ of $T$, identify the connector vertices, $a, b$, and $c$, of a copy of the gadget with the corresponding vertices $t_1, t_2$ and $t_3$ of the graph. The edges of these copies of the gadget are in the edge set of the graph and the interior vertices are in the vertex set. Call the resulting graph $G$.

**Claim 2.3.1.** The set $T$ has a 3-dimensional matching if and only if $G$ has a $\{P_2\}$-factor.
**Proof:** Suppose \( \mathcal{T} \) has a 3-dimensional matching. Then for chosen triples \((t_1, t_2, t_3)\), Figure 2.12 shows the \( \{P_2\} \)-factor of the copy of the gadget having connector vertices \( t_1 = a, t_2 = b \) and \( t_3 = c \).

![Figure 2.12](image)

Figure 2.12: A \( \{P_2\} \)-factor containing the connector vertices

For unchosen triples, that is when \( t_1, t_2, \) and \( t_3 \) are each covered by some other triple, the copy of the gadget having connector vertices \( t_1 = a, t_2 = b \) and \( t_3 = c \), has a \( \{P_2\} \)-factor as shown in Figure 2.13.

![Figure 2.13](image)

Figure 2.13: A \( \{P_2\} \)-factor excluding the connector vertices

Thus, there is always a \( \{P_2\} \)-factor of \( G \) when there is a 3-dimensional matching of \( \mathcal{T} \).

Suppose \( G \) has a \( \{P_2\} \)-factor. Consider a copy of the gadget. If vertex \( a \) is covered by the \( P_2 \) \( x_1x_2a \), then \( a' \) can be covered by choosing \( a'b'y_2 \) or \( a'b'c' \). If we choose \( a'b'y_2 \) then there is no way to cover \( y_1 \). Thus, we must choose \( a'b'c' \) and so \( b \) will be covered by \( y_1y_2b \) and \( c \) by \( z_1z_2c \) as in Figure 2.12. In this case, the triple \((a, b, c)\) is in the 3-dimensional matching.

If instead \( a \) is covered in some other way, then \( x_1 \) must be covered by the \( P_2 \) \( x_1x_2a' \). Thus, \( b' \) can be covered by \( y_1y_2b' \), by \( b'c'z_2 \), or by
In the last case, we are unable to cover vertex $y_1$. Choosing $b'c'z_2$ would leave the vertex $z_1$ uncovered in the packing and so we must choose $y_1y_2b'$. Therefore, $b$ and $c$ will be covered by $P_2$'s other than $y_1y_2b$ and $z_1z_2c$ as in Figure 2.13. In this case, the triple $(a, b, c)$ is not chosen as a triple of the 3-dimensional matching.

The resulting set of chosen triples $T = \{(a, b, c)\}$ is 3-dimensional matching. □

Since the $NP$-complete problem 3DM reduces to the $\{P_2\}$-factor problem, the $\{P_2\}$-factor problem is also $NP$-complete. □

An interesting class of input graphs for which the $\{P_2\}$-packing problem has been studied is the class of square graphs. Let $i \in \{1, \ldots, n\}$ and $j \in \{1, \ldots, m\}$. Consider the grid graph defined by $V = \{1, \ldots, n\} \times \{1, \ldots, m\}$ with $nm$ vertices such that for $i, k \in \{1, \ldots, n\}$ and $j, l \in \{1, \ldots, m\}$, vertex $(i, j)$ is adjacent to vertex $(k, l)$ if and only if $i = k$ and $|j - l| = 1$ or $|i - k| = 1$ and $j = l$. A square graph is a subgraph $G$ of the grid graph with the property that $G$ is connected and each edge of $G$ is contained in some $C_4$ of $G$. An example of a square graph with a $\{P_2\}$-packing is given in Figure 2.14.

![Figure 2.14: A $\{P_2\}$-Packing of a Square Graph](image)

Some results concerning square graphs and their $\{P_2\}$-factors are given in the next theorem.

**Theorem 2.3.2.** [5] Let $G$ be a square graph on $p$ vertices.

(I) If $p \equiv 0 \pmod{3}$, $G$ has a $\{P_2\}$-factor.

(II) If $p \equiv 1 \pmod{3}$ and $v$ is an arbitrary vertex of degree 2 in $G$, $G \setminus v$ has a $\{P_2\}$-factor.
The Family \( \{P_1, P_2\} \)

Results on the packing problem for the family \( \{P_1, P_2\} \) can be found in [3, 1, 2, 5].

An interesting result on \( \{P_1, P_2\} \)-factors, Corollary 2.3.1, concerns triangle graphs. Consider a triangulated grid. A connected subgraph of this grid in which every vertex is contained in a \( K_3 \), is called a triangle graph.

We have the following corollary concerning \( \{P_1, P_2\} \)-factors of triangle graphs. Such a factor is shown in Figure 2.15. This corollary was proven by Akiyama and Kano.

\[ \text{Figure 2.15: A } \{P_1, P_2\} \text{-Factor of a Triangle Graph} \]

**Corollary 2.3.1.** [5] Every triangle graph \( T \) has a \( \{P_1, P_2\} \)-factor.

**Proof:** Let \( S \) be a vertex subset of \( T \). A vertex \( x \) of \( T \) is incident with at most six vertices. Thus, the maximum number of vertices incident with \( x \) which can be isolated in \( T \setminus S \) is four, as shown in Figure 2.16.

\[ \text{Figure 2.16: Maximum isolation of vertices incident with } x \]

Also, each isolated vertex of \( T \setminus S \) must be incident with at least two vertices of \( S \). Therefore,

\[ 2i(T \setminus S) \leq 4|S|. \]
where, \( i(T \setminus S) \) is the number of isolated vertices in \( T \setminus S \).

Thus, by Corollary 2.2.2, with \( k = 2 \), a triangle graph must admit a \( \{P_1, P_2\} \)-factor. \( \square \)

2.4 Families of Cycles

The two-factor problem, which is the \( G \)-factor problem where \( G = \{C_3, C_4, \ldots\} \), is polynomial [22]. In addition to this, the graph packing problem is polynomial for the family \( G = \{C_4, C_5, \ldots\} \), see [31] and for the family \( G = \{C_3, C_5, C_6, \ldots\} \) [7]. The graph packing problem for the family \( G = \{C_5, C_6, \ldots\} \) is expected to be polynomial, see [31], but this problem remains open. For all other finite families of cycles, the packing problem is \( NP \)-complete [29, 31]. The \( NP \)-completeness proofs for the packing problem with families \( G = \{C_6, C_7, \ldots\} \), \( G = \{C_3, C_4, C_6, C_7, \ldots\} \), and \( G = \{C_4, C_6, C_7, \ldots\} \) can be found in [18], and with the family \( G = \{C_3, C_5, C_7, \ldots\} \) can be found in [48].

Two-Factors

A \( \{C_3, C_4, \ldots\} \)-factor, of a graph \( H \) is a subgraph \( F \) of \( H \) for which \( V(F) = V(H) \) and every vertex has degree two. Thus, a two-factor of \( H \) consists of disjoint cycles that cover \( V(H) \).

**Theorem 2.4.1.** [46] A graph \( G = (V, E) \) admits a \( \{C_3, C_4, \ldots\} \)-factor if and only if

\[
|A| \geq \frac{1}{2}(|K| + \chi) \quad \text{for every} \ A \subseteq V
\]

where \( K \) is the set of vertices of degree at most 1 in \( G \setminus A \), and \( \chi \) is the number of components of \( G \setminus A \) which contain an odd number of vertices from \( K \).

2.5 Families of Mixed Graphs

\( \{K_2, F\} \)-Packings

The packing problem where the family \( G \) consists of \( K_2 \) along with some other graph was studied by Cornuéjols, Hartvigsen and Pulleyblank [17], by Hell and Kirkpatrick [28, 36, 29, 30] and by Loebl and Poljak [39, 40]. Loebl and Poljak have completely characterized the complexity of deciding whether a perfect packing exists for such families.
Some early work on the $\{K_2, F\}$-packing problem studied the problem for $F = P_2$ [3, 28, 47] and for $F$ hypomatchable [17, 16, 29]. In both cases the packing problem is polynomial. These results gave hope for other polynomial results. The main polynomial result of this section, Theorem 2.5.1, concerns the class of graphs $F$ which are perfectly matchable, hypomatchable or propellers. A propeller is a graph obtained from a hypomatchable graph $W$ by adding two new vertices, $c$, the centre, and $r$, the root, the edge $cr$ and some edge(s) from $c$ to a nonempty subset of $W$.

**Theorem 2.5.1.** [41, 40] The $\{K_2, F\}$-packing problem is polynomially solvable for all graphs $F$ which are either perfectly matchable, hypomatchable, or a propeller. The problem is $\mathcal{NP}$-complete for all other graphs $F$.

It is clear that if all graphs $F$ are perfectly matchable, then the problem is equivalent to the maximum matching problem. The theorem was proven in [17] for graphs $F$ which are hypomatchable and in [41] for graphs $F$ which are propellers. These two proofs will not be presented here.

## 2.6 Applications

As we have mentioned in Chapter 1, matching theory has many real-world applications. This theory has been used in scheduling, assignment, transportation, network flow, shortest path, travelling salesman and Chinese postman problems.

A specific application for the graph packing problem is the area of examination scheduling. In [27], Hell and Kirkpatrick consider such a problem. The problem of minimizing the number of exam periods and avoiding conflicts such as students taking two exams at a time, is a well-studied problem and is $\mathcal{NP}$-complete. The problem studied in [27] is that of grouping exam periods into sets of $k$ periods. Consider a set of courses which have already been separated into examination periods. The $k$ periods represent $k$ examination days, and the exams in a given period are organized within that day’s schedule. The goal is to minimize the second order conflicts, such as a student being required to sit two exams on the same day or with only a given amount of time between exams. This problem can be formulated in a graph theoretic way.
Instance: A graph $G$, a positive integer $n$ and $\omega$ a weight function on the graph $K_n$ such that $\omega: E(K_n) \rightarrow [0, \infty)$.

Question: Are there $d$ disjoint subgraphs $G_1, G_2, \ldots, G_d$ of $K_n$ such that

1. each $G_i$ is isomorphic to $G$,
2. each vertex of $K_n$ is in some $G_i$, and
3. $\sum_{i=1}^{d} \sum_{e \in E(G_i)} \omega(e)$ is minimized?

In this definition of the problem, the graph $G$, with $k$ vertices, corresponds to the definition of the second order conflicts. The graph $K_n$ represent the $n$ examination periods and we can assume that $n = kd$. The weight function, $\omega(e)$, where $e = (u, v)$, corresponds to the number of students who need to write an exam in both period $u$ and period $v$. Consider the example used in [27]. If $G = K_3$, then $k = 3$ and the solution $G_1, \ldots, G_d$, corresponds to a partition of the examination periods into $d$ days with 3 periods per day. The number of occurrences of a student being required to write two consecutive exams on the same day is minimized.

As we can see from the question stated for this problem, we are looking for a $G$-factor of $K_n$. Notice that in the formulation discussed above, it is assumed that the scheduler has eliminated first order conflicts.
Chapter 3

Directed Graph Packings

Study of the graph packing problem for undirected graphs resulted in many polynomial algorithms as well as some \( \mathcal{NP} \)-completeness results. We now move our focus to the graph packing problem for directed graphs.

After introducing the definitions and notation to be used in this chapter, we will look at the directed graph packing problem for directed paths, cycles and stars.

3.1 Definitions and Notation

Recall from Chapter 1 that a directed graph or digraph \( H = (V(H), A(H)) \) is a graph with the set \( V(H) \) of vertices and the set \( A(H) \) of arcs. Recall that an arc \( ab \) is oriented from \( a \) to \( b \). If we are not interested in the direction, or do not know the direction on a given arc, we will refer to it simply as an edge. A directed \( G \)-packing of a digraph \( H \) is a set of pairwise vertex-disjoint subgraphs of \( H \) each of which is isomorphic to some digraph in the family \( G \). A vertex of \( H \) is covered by the directed \( G \)-packing if it belongs to a subgraph of the \( G \)-packing. If the directed \( G \)-packing has size \( |V(H)| \), then it is a directed \( G \)-factor.

The directed \( G \)-factor problem can be stated as follows:

Given the family of directed graphs \( G \),

**Directed \( G \)-factor:**

- **Instance:** A digraph \( H \).
- **Question:** Does \( H \) admit a directed \( G \)-factor, that is, is there a packing of \( H \) using digraphs from the family \( G \) such that the size of the packing is \( |V(H)| \)?
For the remainder of this chapter, the terms $G$-packing and $G$-factor will refer to directed $G$-packing and directed $G$-factor unless otherwise stated.

As we have mentioned, we will study this problem for directed paths, cycles and stars. A directed path on $k$ edges, denoted $P_k$, is a graph with vertex set $\{v_1, \ldots, v_{k+1}\}$ and arcs $\{(v_i, v_{i+1}) | i \in \{1, \ldots, k\}\}$. A directed cycle on $k$ edges, denoted $C_k$, is a graph with vertices $\{v_1, \ldots, v_k\}$ and arcs $\{v_1v_2, v_2v_3, \ldots, v_kv_1\}$. Finally, a directed star on $k$ edges, denoted $S_k$, is a complete bipartite graph with partitions $\{u\}$ and $\{v_1, \ldots, v_k\}$ and arcs $\{(uv_i) | i \in \{1, \ldots, k\}\}$.

The underlying graph of a directed graph $H$ is the graph $H'$ on the same vertex set as $H$. The edge $uv$ is in $E(H')$ if $uv \in A(H)$ or $vu \in A(H)$ or both.

## 3.2 Packing with Directed Paths

### 3.2.1 One-Element Families

#### The Family $\mathcal{G} = \{\bar{P}_1\}$

**Theorem 3.2.1.** The maximum $\{\bar{P}_1\}$-packing problem is in $\mathcal{P}$.

**Proof:** For a digraph $H$, consider the underlying undirected graph $H'$. Then the $\{\bar{P}_1\}$-packing problem for $H$ becomes the maximum matching problem for $H'$, which is known to be polynomial. Therefore, if we find a maximum matching of the graph $H'$, then by choosing the arcs of $H$ corresponding to the matched edges of $H'$, we obtain a maximum $\{\bar{P}_1\}$-packing of $H$ in polynomial time. $\square$

#### The Family $\mathcal{G} = \{\bar{P}_2\}$

**Theorem 3.2.2.** The $\{\bar{P}_2\}$-factor problem is $\mathcal{NP}$-complete.

**Proof:** The proof of this theorem is analogous to the proof of Theorem 2.3.1. If we have a $\{\bar{P}_2\}$-factor as a certificate, we can verify its correctness in polynomial time. Therefore, the $\{\bar{P}_2\}$-factor problem is in $\mathcal{NP}$. We will now reduce 3DM to the $\{\bar{P}_2\}$-factor problem. Consider the gadget of Figure 3.1.

Recall that an instance of 3DM is a set of triples $\mathcal{T} = \{(t_1, t_2, t_3) | t_1 \in W, t_2 \in X, t_3 \in Y\}$ which is some subset of $W \times X \times Y$ ($W, X, Y = \{1, 2, \ldots, n\}$). We
will construct a graph $G$ such that $\mathcal{T}$ has a 3-dimensional matching if and only if $G$ admits a $\{\vec{P}_2\}$-factor.

Construction: Let the vertex set of $G$ consist of one vertex for each element of $W \cup X \cup Y$. For each triple of $\mathcal{T}$, add a copy of the gadget to the graph by identifying connector vertex $a$ with vertex $t_1$, connector vertex $b$ with $t_2$, and connector vertex $c$ with $t_3$. The interior vertices and the edges of the copy of the gadget become vertices and edges of the graph. Call this graph $G$.

Claim 3.2.1. The set $\mathcal{T}$ has a 3-dimensional matching if and only if $G$ has a $\{\vec{P}_2\}$-factor.

Proof: Suppose $\mathcal{T}$ has a 3-dimensional matching. Then for triples $(t_1, t_2, t_3)$ in the matching we have the $\{\vec{P}_2\}$-factor of the copy of the gadget, having connector vertices $t_1 = a$, $t_2 = b$, and $t_3 = c$, shown in Figure 3.2.
For triples not in the matching, that is when \( t_1, t_2, \) and \( t_3 \) are each covered by some other triple, the corresponding copy of the gadget in \( G \) has a \( \{ \vec{P}_2 \} \)-factor as in Figure 3.3.

![Diagram](image.png)

Figure 3.3: A \( \{ \vec{P}_2 \} \)-factor excluding the connector vertices

Thus, there is always a \( \{ \vec{P}_2 \} \)-factor of \( G \) when there is a 3-dimensional matching of \( T \).

Suppose \( G \) has a \( \{ \vec{P}_2 \} \)-factor. Consider a copy of the gadget. The vertex \( x_1 \) can be covered by choosing the directed path \( x_1x_2a \) or the directed path \( x_1x_2a' \). If it is covered by \( x_1x_2a \), then in order to cover \( a', a'b'c' \) must be in the packing. This in turn forces \( y_1y_2b \) and \( z_1z_2c \) to be in the packing. Thus, if one of the directed paths incident with a connector vertex is in the packing, then the connector vertices of the gadget are covered as in Figure 3.2. In this case, \((a, b, c)\) is chosen as one of the triples in the 3-dimensional matching. If \( x_1 \) had been covered by the directed path \( x_1x_2a' \), then the only way to cover \( b' \) would have been to choose the directed path \( y_1y_2b' \) and this would force \( z_1z_2c' \) to be in the packing as well. Thus if the directed path of a gadget incident with a connector vertex is not covered by the packing, then none of the connector vertices of the gadget will be covered by the packing. In this case, the triple \((a, b, c)\) is not in the 3-dimensional matching.

The resulting set of chosen triples is a 3-dimensional matching. \( \square \)

Thus, the \( \{ \vec{P}_2 \} \)-factor problem is \( NP \)-complete. \( \square \)
CHAPTER 3. DIRECTED GRAPH PACKINGS

The gadget used for the \{\vec{P}_2\}-factor problem can be used for other directed paths of length two by altering the arc directions in the paths \(x_1x_2a, x_1x_2a', a'b'c', y_1y_2b, y_1y_2b', z_1z_2c,\) and \(z_1z_2c'\) accordingly. One such modification of the gadget is used in the \(\mathcal{NP}\)-completeness of the \(S_2\)-packing problem on page 64.

\{\vec{P}_1, \vec{P}_k\}-Factors

The main result of this chapter concerns the graph packing problem with the family \{\vec{P}_1, \vec{P}_2\} and will be given in Section 3.3.

The graph factor problem with the family \{\vec{P}_1, \vec{P}_3\} is actually the same as the \{\vec{P}_1\}-factor problem since any copy of \vec{P}_3 can be perfectly packed with two copies of the graph \vec{P}_1. Therefore, the \{\vec{P}_1, \vec{P}_3\}-factor problem is polynomial. A similar argument shows that all \{\vec{P}_1, \vec{P}_j\}-factor problems where \(j\) is odd are polynomial.

We will now consider the \{\vec{P}_1, \vec{P}_4\}-factor problem. Our approach is based upon work by Loebl and Poljak in [41]. We will consider the \{\vec{P}_1, \vec{P}_4\}-factor problem with a restriction on the vertex set of the input graph. If this restricted problem is \(\mathcal{NP}\)-complete, and we will show that it is, then the \{\vec{P}_1, \vec{P}_4\}-factor problem is also \(\mathcal{NP}\)-complete.

In the restricted \{\vec{P}_1, \vec{P}_4\}-factor problem stated below, the vertex set \(V(G)\) of input graph \(G\) is made up of the disjoint union of a set \(D\) and a set \(A\) of vertices. The set \(D\) is independent, which means that for all \(u, v \in D\), \(uv \notin E(G)\).

**restricted \{\vec{P}_1, \vec{P}_4\}-factor:** Instance: Digraph \(G\) with \(V(G) = D(G) \cup A(G)\), such that \(D\) is independent and \(2|D| = 3|A|\). Question: Does \(G\) admit a \{\vec{P}_1, \vec{P}_4\}-factor?

**Proposition 3.2.1.** A digraph \(G\) with \(V(G) = D(G) \cup A(G)\), such that \(D\) is independent and \(2|D| = 3|A|\) has a \{\vec{P}_1, \vec{P}_4\}-factor if and only if \(G\) admits a \{\vec{P}_4\}-factor \(F\) such that for each of the \(k\) \(\vec{P}_4\)s of \(F\), vertices \(v_1, v_3,\) and \(v_5\) are vertices of \(D\) and vertices \(v_2\) and \(v_4\) are vertices of \(A\).

**Proof:** Consider a digraph \(G\) with \(V(G) = D(G) \cup A(G)\), such that \(D\) is independent and \(2|D| = 3|A|\). If there is a \{\vec{P}_4\}-factor \(F\) of \(G\), with the requirement stated in the proposition, then there is a \{\vec{P}_1, \vec{P}_4\}-factor and there are \(3k\) vertices in \(D\) and \(2k\) in \(A\).
Suppose that we have a \( \{\vec{P}_1, \vec{P}_4\}\)-factor. We will prove by counting that the vertices in \( D \) covered by the factor, are covered by \( \vec{P}_4 \)'s meeting the requirement.

Suppose that the factor uses \( i \vec{P}_4 \)'s such that vertices \( v_1, v_3, \) and \( v_5 \) cover vertices of \( D \) and vertices \( v_2, \) and \( v_4 \) cover vertices of \( A; \) \( j \vec{P}_4 \)'s which cover two vertices of \( D; \) \( n \vec{P}_4 \)'s which cover one vertex of \( D; \) \( p \vec{P}_4 \)'s which cover no vertices of \( D; \) \( l \vec{P}_1 \)'s such that one vertex covers a vertex of \( A \) and the other a vertex of \( D; \) and \( m \vec{P}_1 \)'s such that both vertices cover vertices of \( A. \) Since \( 2|D| = 3|A| \), we have the following:

\[
2(3i + 2j + n + l) = 3(2i + 3j + 4n + 5p + l + 2m)
\]

\[
6i + 4j + 2n + 2l = 6i + 9j + 12n + 15p + 3l + 6m
\]

\[
0 = 5j + 10n + 15p + l + 6m
\]

Since \( j, n, p, l \) and \( m \) are non negative integers, the only solution is that \( j = n = p = l = m = 0 \) and so we only use \( \vec{P}_4 \)'s obeying the requirement in the factor and so we have a \( \{\vec{P}_4\}\)-factor of \( G. \) □

Consider the colouring of a \( \vec{P}_4 \) shown in Figure 3.4.

![Figure 3.4: Colouring of a \( \vec{P}_4 \) used in COL-\( \{\vec{P}_4\}\) ]

We define COL-\( \{\vec{P}_4\} \) as follows:

**COL-\( \{\vec{P}_4\} \):**  
**Instance:** A directed graph \( G \) and a colouring \( \phi_G : V(G) \rightarrow \{0, 1\} \) of the vertices of \( G. \)

**Question:** Does \( G \) admit a \( \{\vec{P}_4\}\)-factor which is faithful to the colouring?

**Theorem 3.2.3.** The COL-\( \{\vec{P}_4\}\)-factor problem is \( \mathcal{NP} \)-complete.

**Proof:** We will reduce 3DM to the COL-\( \{\vec{P}_4\}\)-factor problem. Consider the gadget of Figure 3.5 with connector vertex \( b \) and connector edges \( a \) and \( c. \)

Consider an instance of 3DM for which the sets \( W \) and \( Y \) consist of \( q \) arcs on two vertices each, and the set \( X \) consists of \( q \) vertices.
**Construction:** Consider the vertex set \( V(W) \cup X \cup V(Y) \) and the arc set containing the arcs from sets \( W \) and \( Y \). For each triple \( (t_1, t_2, t_3) \) of \( T \), add a copy of the gadget to the graph by identifying connector arc \( a_1a_2 \) with \( t_1 \), connector vertex \( b \) with \( t_2 \), and connector arc \( c_1c_2 \) with \( t_3t_3' \). The interior arc of these copies of the gadget make up the arc set of the graph. All interior vertices of the gadgets are added to the vertex set. Call the resulting graph \( G' \).

**Claim 3.2.2.** The set \( T \) has a 3-dimensional matching if and only if \( G \) has a COL-{\( \vec{F}_4 \)}-factor.

If \( T \) has a 3-dimensional matching, then for triples in the matching, we have the COL-{\( \vec{F}_4 \)}-factor of the corresponding copy of the gadget shown in Figure 3.6.

**Figure 3.6:** A COL-{\( \vec{F}_4 \)}-factor of the gadget including the connectors

For triples not in the matching, the corresponding copy of the gadget has a COL-{\( \vec{F}_4 \)}-factor as shown in Figure 3.7.
Thus, for any 3-dimensional matching of $\mathcal{T}$, we have a COL-$\{\vec{P}_4\}$-factor of $G$.

Suppose that we have a COL-$\{\vec{P}_4\}$-factor of $G$. Consider the vertex $x$ of a given copy of the gadget. If $x$ is covered by a $\vec{P}_4$ which contains the vertex $t$ but not the vertex $w$, then $w$ cannot be covered. Therefore, $x$ must be covered either by the $\vec{P}_4$ between $w$ and $x$ or by the $\vec{P}_4$ from $b$ to $z$ and then to $y$. Suppose $x$ is covered by the $\vec{P}_4$ from $w$ to $x$. Then $y$ can only be covered if the $\vec{P}_4$ from $z$ to $y$ is chosen. In this case, none of the connectors are chosen and so the triple $(a, b, c)$ is not in the 3-dimensional matching of $\mathcal{T}$.

If $x$ is covered by the $\vec{P}_4$ from $b$ to $z$ to $y$, then $w$ can only be covered by choosing the $\vec{P}_4$ from $w$ to $a$ for the packing and then $z$ can only be covered by choosing the $\vec{P}_4$ from $z$ to $c$. In this case, all connectors will be covered and so we choose the triple $(a, b, c)$ to be in the 3-dimensional matching of $\mathcal{T}$.

This is a 3-dimensional matching of $\mathcal{T}$ since it is generated from a $\{\vec{P}_4\}$-factor.

The COL-$\{\vec{P}_4\}$-factor problem is $\mathcal{NP}$-complete.

Now that we have shown the COL-$\{\vec{P}_4\}$-factor problem to be $\mathcal{NP}$-complete, we can prove the $\mathcal{NP}$-completeness of the $\{\vec{P}_1, \vec{P}_4\}$-factor problem by reduction of the COL-$\{\vec{P}_4\}$-factor problem.

Theorem 3.2.4. The Restricted $\{\vec{P}_1, \vec{P}_4\}$-factor problem is $\mathcal{NP}$-complete.
**Proof:** We will reduce \( \text{COL-}\{\vec{P}_4\} \) to the restricted \( \{\vec{P}_1, \vec{P}_4\}\)-factor problem. Suppose \( G \) is an instance of \( \text{COL-}\{\vec{P}_4\} \). We will create an instance \( G' \) of the restricted \( \{\vec{P}_1, \vec{P}_4\}\)-factor problem in the following way: Firstly, remove edges having both end vertices coloured 1 since these edges are not used in the \( \{\vec{P}_1, \vec{P}_4\}\)-factor problem. All other edges become the edge set of \( G' \). All vertices with colour 1 make up the independent set \( D \) and those with colour 0 make up the set \( A \) of \( V(G') \). If we have a YES-instance of \( \text{COL-}\{\vec{P}_4\} \), the set \( D \) must have size \( 3k \) and the set \( A \) must have size \( 2k \), and all \( \vec{P}_4 \)'s will have vertices from \( D \) and \( A \) as shown in Figure 3.8. Since \( G \) admits a \( \text{COL-}\{\vec{P}_4\}\)-factor, \( G' \) must admit a \( \{\vec{P}_4\}\)-factor. This implies that \( G' \) admits a \( \{\vec{P}_1, \vec{P}_4\}\)-factor.

![Figure 3.8: Placement of \( \vec{P}_4 \)](image)

Now, suppose \( G' \) is a YES-instance of the restricted \( \{\vec{P}_1, \vec{P}_4\}\)-factor problem. Then by Proposition 3.2.1, there is a \( \{\vec{P}_4\}\)-factor \( F \) of \( G' \). If we colour vertices \( v_1, v_3 \) and \( v_5 \) of each \( \vec{P}_4 \) of \( F \) with colour 1 and vertices \( v_2 \) and \( v_4 \) with colour 0, then we have a \( \text{COL-}\{\vec{P}_4\}\)-factor of \( G' \).

**Corollary 3.2.1.** The \( \{\vec{P}_1, \vec{P}_4\}\)-factor problem is \( \mathcal{NP} \)-complete.

**Proof:** Any instance of the restricted \( \{\vec{P}_1, \vec{P}_4\}\)-factor problem is also an instance of the \( \{\vec{P}_1, \vec{P}_4\}\)-factor problem since both problems ask the same question.

Following a similar approach to that used for the \( \{\vec{P}_1, \vec{P}_4\}\)-factor problem, it can be shown that all \( \{\vec{P}_1, \vec{P}_j\}\)-factor problems where \( j \) is even and greater or equal to 4, are \( \mathcal{NP} \)-complete. This again follows from ideas used in [41].

**Infinite Families of Paths**

Stemming from the gadget for the \( \{\vec{P}_2\}\)-packing problem, as well as from gadgets constructed for the \( \{\vec{P}_k\} \), \( \{\vec{P}_2, \vec{P}_k\} \), \( \{\vec{P}_3, \vec{P}_k\} \), and \( \{\vec{P}_4, \vec{P}_k\} \)-packing problems individually, we have been
able to construct a gadget which can be used to prove the \( \mathcal{NP} \)-completeness of families of directed paths \( \vec{P}_k \) with \( k \geq 2 \). Interestingly, our gadget works for all such families except the family \( \{ \vec{P}_2, \vec{P}_3 \} \). However, the different approach taken in [12] allows this case to be treated and it is found to be \( \mathcal{NP} \)-complete as well. Our gadget is constructed in the following way:

**Construction:** Consider a family \( G \) of directed paths. Let \( \vec{P}_k \) be the shortest path in that family. Let \( \vec{P}_l \in G \) and \( \vec{P}_{l+1} \notin G \), where \( 2k - 1 \geq l \geq k + 2 \). Then we construct the gadget shown in Figure 3.9.

![Gadget for path factor problems](image)

Figure 3.9: Gadget for path factor problems

The connector \( b \) is a directed path whose length is \( 2k - l - 1 \). The inequality \( 2k - 1 \geq l \) ensures that the connector \( b \) has at least one vertex. The inequality \( l \geq k + 2 \) ensures that there is at least one vertex between the \( \vec{P}_k \)s \( x_1 \ldots x_{k+1} \) and \( y_1 \ldots y_{k+1} \). This gadget does not work for the \( \{ \vec{P}_2, \vec{P}_3 \} \)-packing problem since the inequality for this case gives \( 3 \geq 3 \geq 4 \).

We have the following theorem:

**Theorem 3.2.5.** The \( G \)-factor problem is \( \mathcal{NP} \)-complete for \( G \subseteq \{ \vec{P}_k, \ldots, \vec{P}_l, \ldots \} \), where \( 2k - 1 \geq l \geq k + 2, k \geq 2, \vec{P}_l \in G \) and \( \vec{P}_{l+1} \notin G \).

**Proof:** We will reduce 3DM to this \( G \)-factor problem. Consider the gadget of Figure 3.9 with connector vertices \( a \) and \( c \) and connector \( b \) a path of length \( 2k - l - 1 \). Consider the 3-dimensional matching problem with \( W \) and \( Y \), two sets having \( q \) points, and \( X \), a set having \( q \) elements, each isomorphic to a path.
of length $2k - l - 1$. For this proof, we consider triples of instances of 3DM to consist of $t_1$ from $W$, $t_3$ from $Y$ and a path $t_2, \ldots, t_{2k-l}$ from $X$.

We construct the graph $G$ as before on the vertex set $W \cup X \cup Y$. For each triple of $T$, we add the arcs and interior vertices of a copy of the gadget to the graph by associating $a$ with $t_1$, $b$ with the $P_{2k-l-1}$, $t_2, \ldots, t_{2k-l}$, and $c$ with $t_3$.

**Claim 3.2.3.** $T$ has a 3-dimensional matching if and only if $G$ has a $G$-factor.

**Proof:** Suppose that $T$ has a 3-dimensional matching. Then for triples in the matching, we have a $G$-factor of the corresponding copies of the gadgets as shown in Figure 3.10.

![Figure 3.10: A $G$-Factor including the connectors](image)

For those triples not in the matching, we have a $G$-factor of the corresponding copies of the gadgets as shown in Figure 3.11.

![Figure 3.11: A $G$-Factor excluding the connectors](image)
Thus, if we have a 3-dimensional matching of $T$, then we have a $G$-factor of the graph $G$.

Suppose that we have a $G$-factor of $G$. Consider one of the copies of the gadget in $G$. Vertex $x_1$ can either be covered by the $P_k x_1 \ldots x_k x_{k+1}$, by the $P_l x_1 \ldots x_{l+1}$, or by the $P_k x_1 \ldots x_k a$. Suppose that it is covered by the first. Then we cannot use the connector $b$. Therefore, the vertices $x_{k+2}, \ldots, x_{l+1}$ cannot be covered by a $P_k$ or by a $P_l$ since the path $x_{k+2} \ldots x_{l+1}$ has length less than $k$. Thus suppose that $x_1$ is covered by the $P_l x_1 \ldots x_{l+1}$. Then in order to cover $y_{k+1}$, we must choose the $P_k y_1 \ldots y_{k+1}$. In this case, none of the connectors are in the $G$-factor and so the triple $(a, b, c)$ is not part of the 3-dimensional matching of $T$.

Suppose that $x_1$ is covered by the $P_k x_1 \ldots x_k a$. Then since $x_{k+1} \ldots x_{l+1}$ and $x_{k+1} \ldots x_{l+1} y_{k+1}$ are both paths of length less than $k$, the only way to cover $x_{k+1}$ is to choose the $P_k b_1 \ldots b_{2k-1} x_{k+1} \ldots x_{l+1} y_{k+1}$. This then forces us to choose the $P_k y_1 \ldots y_k c$ in order to cover the vertices $y_1 \ldots y_k$. In this case, all connectors are covered and so $(a, b, c)$ is part of the 3-dimensional matching of $T$.

Thus, if we have a $G$-factor of $G$, we have a 3-dimensional matching of $T$. □

The $G$-factor problem for $G \subseteq \{P_k \ldots P_l \ldots \}$, $2k - 1 \geq l \geq k + 2$, $k \geq 2$, $P_l \in G$, $P_{l+1} \notin G$, is $NP$-complete. □

### 3.3 Main Theorem

This theorem makes use of augmenting configurations. An augmenting configuration of a graph $H$ with respect to a packing $P$ of $H$ is a special subgraph of $H$ such that the number of vertices covered by $P$ in the subgraph can be increased, or augmented, by removing certain edges from $P$ and adding other edges to $P$, or simply by adding edges to $P$. Recall that an arc $ab$ is a directed edge beginning at vertex $a$ and ending in vertex $b$. If we do not know the direction of an arc or if the direction is not important we will simply refer to it as the edge $ab$. A walk is a sequence $v_0 e_1 v_1 e_2 \ldots e_k v_k$ of vertices and edges such that $e_i = v_{i-1} v_i$
for all $i$. If no edges of the sequence are repeated, then we have a \textit{trail}. We will simply give the vertices of the trail in this proof. There are directed edges between consecutive vertices of the trail. All of our augmenting configurations are trails. Note that a vertex may be repeated in a trail. In this case, the augmenting configuration is \textit{self-intersecting}.

Theorem 3.3.1 states our main result: a $\{\vec{F}_1, \vec{F}_2\}$-packing is maximum if and only if there are no augmenting configurations. A polynomial time algorithm for the $\{\vec{F}_1, \vec{F}_2\}$-packing problem can be found in [12]. This algorithm is inspired by Hall's theorem and the resulting algorithm for matchings in bipartite graphs. In the remainder of this section, the term packing will be used to refer to a $\{\vec{F}_1, \vec{F}_2\}$-packing.

In this chapter, an \textit{alternating trail} is a trail $u_0, v_0, u_1, v_1, \ldots, v_{k-1}, u_k$ with an even number of edges, where $u_0$ is the start vertex, an exposed vertex of $\mathcal{P}$. For each $i \in \{0, 1, \ldots, k-1\}$ we have the following:

- $u_iv_i$ is an edge of $H$ which is not in the packing $\mathcal{P}$.
- $v_iu_{i+1}$ is an edge of the packing $\mathcal{P}$.
- The edges $u_iv_i$ and $v_iu_{i+1}$ are either both oriented towards the vertex $v_i$ or both oriented away from the vertex $v_i$.
- $u_{i+1}$ is incident with exactly one arc of $\mathcal{P}$ and $v_i$ is incident with one or two arcs of $\mathcal{P}$.

Note that if $v_i$ is incident with two arcs of $\mathcal{P}$, then one of those arcs is directed towards $v_i$ and one is directed away from $v_i$. Also, no vertex $u_i$ can appear twice in an alternating trail.

An augmenting configuration is an alternating trail of one of the following types:

1. Type 1: A trail $u_0, v_0, u_1, v_1, \ldots, u_k, z$, where $u_0, v_0, u_1, \ldots, u_k$ is an alternating trail $T$ and $z$ is an exposed vertex of $\mathcal{P}$.

2. Type 2: A trail $u_0, v_0, u_1, v_1, \ldots, u_k, z$, where $u_0, v_0, u_1, \ldots, u_k$ is an alternating trail $T$ and $z = v_i$, $0 \leq i \leq k - 1$. Then $u_kv_iu_i$ is a $\vec{F}_2$ but is not in $\mathcal{P}$ and $v_iu_{i+1}$ in $\mathcal{P}$ is not part of a $\vec{F}_2$ of $\mathcal{P}$, or

(a) $z = v_i$, $0 \leq i \leq k - 1$. Then $u_kv_iu_i$ is a $\vec{F}_2$ but is not in $\mathcal{P}$ and $v_iu_{i+1}$ in $\mathcal{P}$ is not part of a $\vec{F}_2$ of $\mathcal{P}$, or

(b) $z = u_i$, $0 \leq i \leq k - 1$. Then $u_kv_iu_i$ is a $\vec{F}_2$ but is not in $\mathcal{P}$ and $v_iu_{i+1}$ in $\mathcal{P}$ is not part of a $\vec{F}_2$ of $\mathcal{P}$.
3. Type 3: A trail $u_0, v_0, u_1, v_1, \ldots, u_k, y, z$, where $u_0, v_0, u_1, \ldots, u_k$ is an alternating trail. $u_ky$ is not in $\mathcal{P}$ and $yz$ is in $\mathcal{P}$.

(a) $u_kyz$ is a $\tilde{P}_2$, or

(b) $z = v_i, 0 \leq i \leq k - 1$. Then $yv_iu_{i+1}$ is a $\tilde{P}_2$ in $\mathcal{P}$.

4. Type 4: A trail $u_0, v_0, u_1, v_1, \ldots, u_k, x, y, z$, where $u_0, v_0, u_1, \ldots, u_k$ is an alternating trail. $u_kx$ is not in $\mathcal{P}$ and $xyz$ is a $\tilde{P}_2$ of $\mathcal{P}$.

The augmenting configurations of type 1, 3a, and 4 are not self-intersecting. Recall that this means no vertex is encountered more than once in the trail. Augmenting configurations of type 2a, 2b and 3b are self-intersecting. Examples of each type of augmenting configuration are given in Figure 3.12. In this figure, dotted lines are edges which are in the packing $\mathcal{P}$ and solid lines are edges which are not in $\mathcal{P}$. Circled vertices are exposed in $\mathcal{P}$.

In all of these types of augmenting configurations, $u_0$ is an exposed vertex which we call the start component, any sequence of vertices $u_i;v_iu_{i+1}$ is called a middle component and $x$, $y$ and $z$ are end components. These components are represented in Figure 3.13.
<table>
<thead>
<tr>
<th>Configuration</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type 1</td>
<td><img src="image1" alt="Type 1 Example" /></td>
</tr>
<tr>
<td>Type 2a</td>
<td><img src="image2" alt="Type 2a Example" /></td>
</tr>
<tr>
<td>Type 2b</td>
<td><img src="image3" alt="Type 2b Example" /></td>
</tr>
<tr>
<td>Type 3a</td>
<td><img src="image4" alt="Type 3a Example" /></td>
</tr>
<tr>
<td>Type 3b</td>
<td><img src="image5" alt="Type 3b Example" /></td>
</tr>
<tr>
<td>Type 4</td>
<td><img src="image6" alt="Type 4 Example" /></td>
</tr>
</tbody>
</table>

Figure 3.12: Examples of augmenting configurations for the $\{\overrightarrow{P_1}, \overrightarrow{P_2}\}$-packing problem
Start components

\[ \begin{align*}
&\text{Type 1:} & u_1 & \xrightarrow{\epsilon} & v_1 & \xrightarrow{\epsilon} & u_{i+1} \\
&\text{Type 2:} & u_1 & \xrightarrow{\epsilon} & v_1 & \xrightarrow{\epsilon} & u_{i+1}
\end{align*} \]

Middle components

\[\begin{align*}
&\text{Type 1:} & u_1 & \xrightarrow{\epsilon} & v_1 & \xrightarrow{\epsilon} & u_{i+1} \\
&\text{Type 2:} & u_1 & \xrightarrow{\epsilon} & v_1 & \xrightarrow{\epsilon} & u_{i+1}
\end{align*}\]

End components

\[\begin{align*}
&\text{Type 1:} & z & \xrightarrow{\epsilon} & z \\
&\text{Type 2:} & z & \xrightarrow{\epsilon} & z \\
&\text{Type 3a:} & y & \xrightarrow{\epsilon} & z & \xrightarrow{\epsilon} & y \\
&\text{Type 3b:} & y & \xrightarrow{\epsilon} & z & \xrightarrow{\epsilon} & y \\
&\text{Type 4:} & x & \xrightarrow{\epsilon} & y & \xrightarrow{\epsilon} & z & \xrightarrow{\epsilon} & x \\
\end{align*}\]

* denotes that this vertex may have been previously encountered

Figure 3.13: Augmenting configuration components for the \{P_1, \tilde{P}_2\}-packing problem
CHAPTER 3. DIRECTED GRAPH PACKINGS

Now we will look at how augmenting configurations are augmented. Augmented configurations for each example of Figure 3.12 are given in Figure 3.14.

<table>
<thead>
<tr>
<th>Configuration</th>
<th>Augmented example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type 1</td>
<td><img src="image1" alt="Type 1 Configuration" /></td>
</tr>
<tr>
<td>Type 2a</td>
<td><img src="image2" alt="Type 2a Configuration" /></td>
</tr>
<tr>
<td>Type 2b</td>
<td><img src="image3" alt="Type 2b Configuration" /></td>
</tr>
<tr>
<td>Type 3a</td>
<td><img src="image4" alt="Type 3a Configuration" /></td>
</tr>
<tr>
<td>Type 3b</td>
<td><img src="image5" alt="Type 3b Configuration" /></td>
</tr>
<tr>
<td>Type 4</td>
<td><img src="image6" alt="Type 4 Configuration" /></td>
</tr>
</tbody>
</table>

Figure 3.14: Examples of augmented configurations for the \( \{\vec{P}_1, \vec{P}_2\} \)-packing problem

Consider the type 1 configuration. To augment, remove edges \( v_i u_{i+1}, i = 0, \ldots k-1 \) from \( \mathcal{P} \) and add edges \( u_i v_i, i = 0, \ldots, k-1 \) and \( u_k z \) to \( \mathcal{P} \). For the type 2a and 2b configurations, remove edges \( v_i u_{i+1}, i = 0, \ldots k-1 \) from \( \mathcal{P} \) and add edges \( u_i v_i, i = 0, \ldots, k-1 \) and \( u_k z \) to \( \mathcal{P} \). For a type 3a configuration, remove edges \( v_i u_{i+1}, i = 0, \ldots k-1 \) from \( \mathcal{P} \) and add edges \( u_i v_i, i = 0, \ldots, k-1, u_k y \) to \( \mathcal{P} \) and keep edge \( yz \) in \( \mathcal{P} \). To augment the type 3b configuration, remove edges \( v_i u_{i+1}, i = 0, \ldots k-1 \) and \( yz \) from \( \mathcal{P} \) and add edges \( u_i v_i, i = 0, \ldots, k-1 \) and
$u_ky$ to $P$. Finally, to augment type 4 configurations, remove edges $v_iu_{i+1}$, $i = 0, \ldots, k - 1$ and $xy$ from $P$, add edges $u_iv_i$, $i = 0, \ldots, k - 1$ and $u_kx$ to $P$ and keep edge $yz$ in $P$.

In Theorem 3.3.1, we will use the notation $G \setminus ab$, where $ab \in E(G)$ to represent the subgraph of $G$ having the same vertex set as $G$ and with edge set $E(G \setminus ab) = E(G) \setminus ab$.

**Theorem 3.3.1.** A $\{\vec{F}_1, \vec{F}_2\}$-packing $P$ of a directed graph $G$ is of maximum size if and only if there exists no augmenting configuration.

**Proof:** Suppose $G$ contains an augmenting configuration with respect to $P$. Then clearly by augmenting we will increase the size of $P$ and so it could not have been of maximum size.

Suppose that $P$ is not of maximum size. Let $P'$ be a larger $\{\vec{F}_1, \vec{F}_2\}$-packing of $G$. We will prove that $P$ must admit an augmenting configuration.

We prove the following statement by induction on $|E(G)|$, the number of edges in the graph $G$:

*For all $k$, if $G$ has $k$ more vertices covered by $P'$ than by $P$, then there are $k$ vertices which start an augmenting configuration in $G$ with respect to $P$."

Consider the case $|E(G)| = 0$. In this case, $k = 0$ and so we have 0 augmenting configurations and the statement holds.

Suppose that the statement holds for all $k$ for any graph with fewer than $|E(G)|$ edges. Suppose that $G$ has $\{\vec{F}_1, \vec{F}_2\}$-packings $P$ and $P'$ such that there are $k \geq 1$ more vertices covered by $P'$ than by $P$. We will use the terminology $P$-edge and $P'$-edge to denote that an edge is a $\vec{F}_1$ of $P$, or of $P'$, respectively. Similarly $P$-path and $P'$-path, will denote a $\vec{F}_2$ which is in the packing $P$, or of $P'$ respectively. Note we can assume $E(G) = P \cup P'$. If not, consider the spanning subgraph of $G$ with edge set $P \cup P'$ and apply our statement. The $k$ vertices which begin augmenting configurations in this spanning subgraph also begin augmenting configurations in $G$. Let $v$ be a vertex covered only by $P'$.

Then we have the following three cases to consider:

1. The vertex $v$ is covered by a $P'$-edge $vw$.
2. The vertex $v$ is covered by a $P'$-path $vwu$.
3. The vertex $v$ is covered by a $P'$-path $w_1vw_2$. 
We will now treat each case with its accompanying subcases. All these proofs follow a similar format. We will look at a subgraph $G \setminus ab$ of $G$ and we will prove that $k$ vertices start augmenting configurations of $\mathcal{P}$ restricted to $G \setminus ab$ and these are either preserved or can be modified to produce augmenting configurations in $G$. Note that in the following proofs, the term augmenting configuration will always refer to an augmenting configuration with respect to the packing $\mathcal{P}$.

1. Suppose vertex $v$ is covered by a $\mathcal{P}'$-edge $vw$.

   (a) Suppose $w$ is also exposed in $\mathcal{P}$. Consider the graph $G \setminus vw$ and the packings $\mathcal{P}$ and $\mathcal{P}'$ restricted to $G \setminus vw$. In this graph, there are $k - 2$ vertices covered by $\mathcal{P}'$ and not by $\mathcal{P}$ and so by our statement, there are $k - 2$ vertices which begin augmenting configurations in $G \setminus vw$. Since $v$ and $w$ are exposed with respect to both $\mathcal{P}$ and $\mathcal{P}'$ in $G \setminus vw$, they cannot appear in any augmenting configurations. Consequently, none of the $k - 2$ vertices beginning augmenting configurations are affected by the edge $vw$ and so all augmenting configurations are preserved in $\mathcal{P}$. Clearly, both $v$ and $w$ begin augmenting configurations in $G$, namely $vw$ and $wv$, and so we have $k$ vertices which begin augmenting configurations in $G$.

   (b) Suppose $w$ is covered by a $\mathcal{P}$-edge $wu$ and $wu \notin \mathcal{P}'$. Consider the graph $G \setminus wu$. In this graph, $\mathcal{P}'$ covers $k + 2$ more vertices than $\mathcal{P}$. Vertices $v$ and $w$ begin augmenting configurations, $vw$ and $wv$, in $G \setminus wu$. Since neither $v$ nor $w$ are incident with any $\mathcal{P}$-edges or $\mathcal{P}$-paths in $G \setminus wu$, these are the only two augmenting configurations in which they appear. Therefore, there are $k$ other vertices which begin augmenting configurations and we need only consider those augmenting configurations which contain $u$ since all augmenting configurations which do not involve $u$ are augmenting configurations of $\mathcal{P}$ in $G$ as well. The exposed vertex $u$ must either start or end an augmenting configuration. Suppose there is an augmenting configuration of type 1 $u_0v_0u_1\ldots u_kz$, where $z = u$. If $u_0u_kw$ is a $\tilde{P}_2$, then $u_0v_0\ldots u_kw$ is an augmenting configuration of type 3a in $G$. Otherwise, $u_0u_kw$ is a middle component and then $u_0v_0\ldots u_kwv$ is an augmenting configuration of type 1 in
C\AP\ER 3. \DIRECTED GRAPH PACKINGS

At this point, we know that any vertex \(u_0 \neq u\) which starts an augmenting configuration of \(P\) in \(G \setminus wu\) which does not contain \(u\) or which ends at \(u\) also starts an augmenting configuration in \(G\). We also know that there are at least \(k - 1\) such vertices. If there are \(k\), then the proof is complete. If not, then the \(k\)th start vertex must be \(u\).

Suppose \(u\) begins an augmenting configuration \(uv_0u_1 \ldots u_kE, E = z, yz,\) or \(zyz,\) in \(G \setminus wu\). If \(vwu\) is a \(P_2\), then \(vwu\) is an augmenting configuration of type \(3a\) starting at \(v\) in \(G\). Otherwise, \(vwuv_0 \ldots u_kE\) is an augmenting configuration in \(G\). This gives the \(k\)th vertex which begins an augmenting configuration in \(G\).

(c) Suppose that \(w\) is on a \(P\)-path \(wuu'\). Consider the graph \(G \setminus wu\). In this graph, \(P'\) covers \(k + 1\) more vertices than \(P\). As before, \(vw\) and \(uw\) are augmenting configurations so we will turn our attention to the remaining \(k - 1\) vertices, other than \(v\) and \(w\), which begin augmenting configurations in \(G \setminus wu\). Since \(u\) and \(u'\) are covered by a \(P\)-edge in \(G \setminus wu\), neither can start a configuration. Consider an augmenting configuration which begins at a vertex \(u_0\). Suppose that \(uu'\) is a \(P\)-edge in this configuration. Note that \(u\) and \(u'\) cannot appear on a \(P\)-edge other than \(uu'\) in an augmenting configuration since \(uu'\) belongs to the \(P\)-path \(wuu'\) in \(G\) and this would imply that one of them was incident with three \(P\)-edges in \(G\) or was on a path of \(P\) of length three. Since the argument for such a configuration, with \(uu'\) a \(P\)-edge in \(G \setminus wu\) and part of a \(P\)-path in \(G\), will be used frequently throughout the proof of Theorem 3.3.1, we will prove the following lemma to which we will refer throughout the remainder of the proof.

**Lemma 3.3.1.** Let \(G\) be a graph, \(P\) a \(\{\tilde{P}_1, \tilde{P}_2\}\)-packing of \(G\) and let \(ab\) be a \(P\)-edge of some \(P\)-path. If \(G \setminus ab\) has an augmenting configuration of \(P\) beginning at \(u_0, u_0 \neq a, b\), then either there is an augmenting configuration beginning at \(u_0\) in \(G\) or there is an augmenting configuration of type \(2a\) with \(a = z\) and \(b\) not in the configuration.

**Proof:** Suppose \(a\) is the middle vertex of the \(P\)-path \(a'ab,\) otherwise reverse the roles of \(a\) and \(b\). In the graph \(G \setminus ab,\) if \(b\) is an exposed vertex in an augmenting configuration beginning
at a vertex \( u_0 \), which does not contain \( a' \) or \( a \). then \( b = z \), and so in \( G \) we would have an augmenting configuration \( u_0 \ldots baa' \) of type 4.

If vertex \( a \) is covered by some \( \mathcal{P} \)-edge \( a'a \) in some augmenting configuration \( u_0v_0u_1 \ldots v_{k-1}u_kE \) in \( G \setminus ab \), where \( E = z, yz \) or \( xyz \), then either \( b \) is in the configuration or it is not. If \( b \) is in the configuration, then \( b = z \) and we have an augmenting configuration of \( \mathcal{P} \) in \( G \) of type 3b with \( a = v_j \) and \( a' = u_{j+1} \).

If \( b \) is not in the configuration, then \( a \) is a \( u_i \), \( 0 < i \leq k \) or a \( v_i \), \( 0 \leq i < k \). If \( a = u_i \) in any type of configuration, then \( a' = v_{i-1} \) and in \( G \) we have a type 4 ending \( a'ab \) to the configuration.

It remains to consider the case where \( a = v_i \). If \( a = v_i \) and in the trail from \( u_0 \) to \( a \) no vertex is revisited in the augmenting configuration, that is, we have configurations of type 1, 3a or 4 or we are in the part of configurations 2a, 2b and 3b between \( u_0 \) and the revisited vertex, then \( a' = u_{i+1} \) and in \( G \) the \( \mathcal{P} \)-edge \( ab \) is part of a middle of this augmenting configuration and so the augmenting configuration is preserved in \( G \).

Now suppose \( a = v_i \) and the vertex \( u_j \) or \( v_j, j \leq i \) is revisited in the augmenting configuration. The possible configurations are those of types 2a, 2b and 3b. First, consider an augmenting configuration of type 2a. Either \( i = j \) or \( i > j \). If \( a = v_i \), and \( i = j \), then we have the second result of the lemma. If \( a = v_i \), and \( i > j \), then in \( G \) this augmenting configuration will be preserved, the \( \mathcal{P} \)-edge \( ab \) will just be part of a middle.

Suppose that the augmenting configuration \( u_0v_0 \ldots u_kz \) is of type 2b. Then vertex \( u_j \) is revisited and for \( a = v_i \), we have \( i > j \) and \( a' = u_{i+1} \) in the alternating trail \( u_0v_0 \ldots u_k \). In \( G \) we can find a new augmenting configuration beginning at \( u_0 \) in the following way. Travel along the alternating trail from \( u_0 \) to \( u_j \). Instead of continuing to \( v_j \), follow the edge \( u_ju_k \).

If \( u_ju_kv_{k-1} \) is a \( \overline{P}_2 \), then we have encountered an augmenting configuration \( u_0v_0 \ldots u_ju_kv_{k-1} \) of type 3a. Otherwise, we have
a middle $u_j u_k v_{k-1}$. In this case, if $v_{k-1}$ is the middle vertex of a $P$-path, then we have an ending of type 4. If not, we have a middle $u_j u_k v_{k-1}$ and we look at the next edge covered in $P$, $v_{k-2} u_{k-1}$ on the alternating trail. Again, we will have an ending of type 3a, or of type 4 or we will have a middle. If we have a middle, continue to the next edge. We will find an ending since the vertex $a = v_i$ of the edge $aa'$ is the middle vertex of the $P$-path $a'ab$.

Finally, suppose that an augmenting configuration of type 3b is found in $G \setminus ab$. Vertex $a$ could never equal $v_j$, where $v_j$ is the revisited vertex, since this would mean that $a$ was incident with three $P$-edges in $G$. Thus, $a = v_i$ and $a' = u_{i+1}$, $i > j$, and in $G$ this augmenting configuration is preserved since $ab$ will just be part of a middle of the augmenting configuration.

We now return to case 1c where $w$ is on a $P$-path $ww'$. Note that for a configuration of type 2a in $G \setminus wu$, if $u = v_i = z$, then in $G$ we have the augmenting configuration $u_0 v_0 \ldots uu' \ldots uwv$. Augment this configuration as follows: All arcs of the augmenting configuration which were covered in $P'$ become covered in $P$ and uncovered in $P'$, and all arcs of the augmenting configuration which were covered in $P$ become covered in $P'$ and uncovered in $P$.

All augmenting configurations of $P$ found in $G \setminus wu$ are preserved or can be modified to produce new augmenting configurations, beginning at the same vertex $u_0$, in $G$, as shown in Lemma 3.3.1. Therefore, there are $k - 1$ vertices which begin augmenting configurations in $G$ and which do not begin at $v$ or $w$. The augmenting configuration $vwwuu'$ gives us $k$ vertices which begin augmenting configurations in $G$.

(d) Suppose $w$ is the middle vertex of a $P$-path $u'wu$, oriented from $u'$ to $u$. Either $vw$ and $u'w$ form a middle component or $vw$ and $uw$ form a middle component. Say it is $vw$ and $u'w$, otherwise reverse the name of $u$ and $u'$. Consider the graph $G \setminus u'w$. Since the vertex $u'$ is exposed in $G \setminus u'w$, $P'$ covers $k + 1$ more vertices than $P$. Therefore there are
at least $k$ vertices other than $u'$ which start augmenting configurations in $G \setminus u'w$. By Lemma 3.3.1 all are preserved in $G$ except possibly a type 2a configuration with $w = v_{j+1}$ and $u = u_{j+1}$. However, such a configuration cannot exist since this would mean that $w$ was incident with three $P'$-edges in $G$.

2. Suppose vertex $v$ is covered by a $P'$-path $vuw$.

(a) Suppose $w$ is exposed in $P$ and consider the graph $G \setminus vw$. The packing $P'$ covers $k - 1$ more vertices than $P$ in $G \setminus vw$ and so there are $k - 1$ vertices which begin augmenting configurations in $G \setminus vw$ and all of these configurations are also configurations in $G$. The reason for this is that adding the $P'$-edge $vw$ to any augmenting configuration cannot destroy the configuration since $w$ is exposed in $G \setminus vw$. Since $vw$ is an augmenting configuration of $G$ beginning at $v$, we have $k$ vertices which begin augmenting configurations in $G$.

(b) Suppose that $w$ and $u$ are covered in $P$ by (i) the $P$-edge $wu$, or (ii) a $P$-path $wuu'$, or (iii) a $P$-path $uu'w$. Examples of these cases are shown in Figure 3.15. Note that in these diagrams, the edge $wu$ is in both $P$ and $P'$.

![Figure 3.15: Example diagrams for case 2b](image)

For cases (i) and (ii), consider the graph $G \setminus vw$. In this graph, $P'$ covers $k - 1$ more vertices than $P$. Since no vertices covered in $P$ have become exposed in $P$ through the restriction to $G \setminus vw$, any augmenting configuration beginning at $u_0$ involving $w$, $u$ or $u'$ in $G \setminus vw$ is still an augmenting configuration beginning at $u_0$ in $G$ since adding $P'$-edges cannot destroy augmenting configurations.

In case (i), $vwu$ is an augmenting configuration of type 3a in $G$, and in
case (ii), $vwu'$ is an augmenting configuration of type 4 in $G$ and so in both cases, there are $k$ vertices which begin augmenting configurations in $G$.

For case (iii), consider the graph $G \setminus wu'$. In this graph, $P'$ covers $k + 1$ more vertices than $P$. The vertex $u'$ is exposed with respect to $P$ in $G \setminus wu'$. If $u'$ begins an augmenting configuration, then in $G$, this augmenting configuration will be destroyed. Therefore, there are $k$ other vertices which begin augmenting configurations in $G \setminus wu'$. The rest of this proof follows from Lemma 3.3.1. Note that we cannot have an augmenting configuration of type 2a with $w = r$ and $u = u_{j+1}$ since this would imply that $w$ had degree three with respect to $P'$ in $G$. Thus there are $k$ vertices which begin augmenting configurations in $G$.

(c) Suppose that $wu$ is not in $P$. Then $w$ can be covered by (i) a $P$-edge $aw$, or (ii) a $P$-path $a'aw$, or (iii) a $P$-path $awa'$. Examples of these cases are shown in Figure 3.16.

![Figure 3.16: Example diagrams for case 2c](image)

In cases (i) and (ii), consider the graph $G \setminus aw$. In case (iii), if $vwa'$ is a $\bar{P}_2$, consider $G \setminus aw$. Otherwise, $vwa$ is a $\bar{P}_2$ and then consider $G \setminus a'w$. In case (i), $P'$ covers $k + 2$ more vertices than $P$. In cases (ii) and (iii), $P'$ covers $k + 1$ more vertices than $P$. In cases (i) and (ii), $vw$ and $wu$ are augmenting configurations and so there are $k$ and $k - 1$ other vertices respectively which begin augmenting configurations in $G \setminus aw$. Consider case (i). If an augmenting configuration contains $a$, then it either begins or ends in $a$. If it begins at $a$, then we can construct an augmenting configuration beginning at $v$ as we did in case 1b. Suppose an augmenting configuration begins at some $u_0 \neq a$ and ends at $a$.
with the edge \( u_k a \). If \( u_k a \) and \( aw \) form a \( \vec{P}_2 \), then \( u_0 \ldots u_k aw \) is an augmenting configuration of type 3a in \( G \). Otherwise, \( awv \) provides an ending of type 1 for the augmenting configuration in \( G \).

Suppose an augmenting configuration ends at vertex \( w \). Then it must be an augmenting configuration \( u_0v_0 \ldots uw \) of type 1. If \( uwa \) is a \( \vec{P}_2 \), then \( uwa \) is a type 3a configuration in \( G \). Otherwise, \( uwa \) must form a middle. In this case, we consider vertex \( a \) which either begins an augmenting configuration or does not. If \( a \) begins an augmenting configuration \( a \ldots u_k E \), then consider \( u_0v_0 \ldots uw \) and \( a \ldots u_k E \), where \( a \neq u_0 \). If no vertex of \( a \ldots u_k E \) appears in \( u_0v_0 \ldots uw \), then \( u_0v_0 \ldots uwa \ldots u_k E \) is an augmenting configuration of \( G \). If a vertex is repeated, then it is a \( v_i \), \( 0 \leq i < k \), or a \( u_i \), \( 0 < i \leq k \). If vertex \( v_i \) is the first vertex of \( a \ldots u_k E \) which is also in \( u_0v_0 \ldots uw \), then \( u_0 \ldots uwa \ldots v_i \) is an augmenting configuration of type 2a in \( G \) provided that \( v \) is not the middle vertex of a \( P \)-path. If \( v_i \) is the middle vertex of a \( P \)-path, then consider \( u_0 \ldots v_i \ldots wa \ldots v_i \ldots u_k E \) in \( G \). First augment between \( v_i \) and \( E \) by switching along the path \( v_i \ldots u_k \) and augmenting the ending \( E \). Then \( u_0 \ldots v_i \ldots wa \ldots v_i \) is an augmenting configuration of type 2a in \( G \). If a vertex \( u_i \) is the first vertex of \( a \ldots u_k E \) which is also in \( u_0v_0 \ldots uw \), then \( u_0 \ldots uwa \ldots u_i \) is an augmenting configuration of type 2b in \( G \).

Now consider \( u_0w_0 \ldots uw \) and \( a \ldots u_k E \), where \( a = u_0 \). Then there is one augmenting configuration which begins at \( a \) and one which ends at \( w \). This is simply an augmenting configuration which begins at \( a \) and ends at \( w \). Then we can construct an augmenting configuration as in 1b and we preserve our \( k \) vertices which start augmenting configurations.

Now suppose we have an augmenting configuration \( u_0v_0 \ldots uw \) of type 1, \( uwa \) is a middle, and \( a \) does not begin an augmenting configuration in \( G \setminus aw \). We claim that it is possible to modify so that only one augmenting configuration ends at \( w \). To see this, suppose that two augmenting configurations end at \( w \), one beginning at a vertex \( u_0 \) and the other at a vertex \( c \). Both must end with the \( P' \)-edge \( uw \). Now either \( u \) is the first vertex common to both augmenting configurations or it is not. If it is, then the augmenting configuration beginning at \( u_0 \)
has a $P$-edge $l_1u$ and the augmenting configuration beginning at $c$ has a $P$-edge $l_2u$. The edges $l_1u$ and $l_2u$ must form a $\bar{P}_2$ and so one of the augmenting configurations, say the one beginning at $c$ can be modified to an augmenting configuration $c \ldots l_2u l_1$ of type 4. Then there is just one augmenting configuration ending at $w$.

If $u$ is not the first intersection of the two augmenting configurations, then the first intersection is at some vertex $v_i$, $0 \leq i < k$, or $u_i$, $0 < i \leq k$. Note that if $v_0$ is the first intersection, the $u_0v_0c$ is a $P'$-path and so either $u_0v_0u_1$ or $cv_0u_1$ is a $\bar{P}_2$ and hence an augmenting configuration of type 3a which does not end at $w$. If the augmenting configurations first intersect at a $u_i$, then either $l_1u_i$ is a $P$-edge of $u_0 \ldots l_1u_i \ldots w$ and $l_2u_i$ is a $P$-edge of $c \ldots l_2u_i \ldots w$, or $l_1u_i$ is a $P$-edge and $l_2u_i$ is a $P'$-edge. In the first case, $l_1u_i l_2$ is a $P$-path and so one augmenting configuration, say the one beginning at $c$ can be modified to become the augmenting configuration $c \ldots l_2u l_1$ of type 4, and the augmenting configuration beginning at $u_0$ and ending at $w$ is preserved. In the second case, we have $l_2u_i v_i$ which is a $P'$-path, not a middle and so the augmenting configuration beginning at $c$ is not legitimate.

If the augmenting configurations first intersect at some vertex $v_i$, then either $l_1v_i$ and $l_2v_i$ are $P'$-edges or else $l_1v_i$ is a $P'$-edge and $l_2v_i$ is a $P$-edge. In the first case, either $l_1v_i u_{i+1}$ or $l_2v_i u_{i+1}$ is a $\bar{P}_2$ and so the corresponding augmenting configuration is actually of type 3a with ending $l_1v_i u_{i+1}$ or $l_2v_i u_{i+1}$ and so there is only one augmenting configuration ending at $w$. In the second case, $l_2v_i u_{i+1}$ must be a $\bar{P}_2$ and so $c \ldots l_2v_i u_{i+1}$ is an augmenting configuration of type 4 and only one augmenting configuration ends at $w$.

Therefore, only one augmenting configuration ends at $w$. If $uwa$ is a $\bar{P}_2$, then this augmenting configuration acquires the ending $uwa$ of type 3a in $G$. Otherwise, this augmenting configuration is destroyed by the addition of $aw$. However, in this case, $vwa$ is an augmenting configuration beginning at vertex $v$ and so we still have $k$ vertices which begin augmenting configurations in $G$ and we are done.
Case (ii) follows from Lemma 3.3.1. In the case of an augmenting configuration of type 2a with \( a = v_j = z \), and \( a' = u_{j+1} \), the augmenting configuration is destroyed in \( G \) but \( vwa'a' \) is an augmenting configuration of \( G \) and so we still have \( k \) vertices beginning augmenting configurations.

In case (iii), if we are considering \( G \setminus aw \), then \( vwa' \) is an augmenting configuration of \( G \). Otherwise, \( vwa \) is an augmenting configuration of \( G\setminus a'w \). In either case, there are \( k \) other vertices which start augmenting configurations in \( G \setminus aw \), or \( G \setminus a'w \). Case (iii) then follows from Lemma 3.3.1. An augmenting configuration of type 2a not involving \( v \) with \( w = v_j = z \), and \( a' = u_{j+1} \) (or \( a = u_{j+1} \) if we have \( G \setminus a'w \)), cannot exist since \( w \) cannot be incident with three \( P' \)-edges. If the augmenting configuration does start at \( v \), then it is destroyed in \( G \) and we use the augmenting configuration \( vwa' \) (or \( vwa \)) instead.

3. Suppose vertex \( v \) is covered by a \( P' \)-path \( w_1vw_2 \), oriented from \( w_1 \) to \( w_2 \).

If either \( w_1 \) or \( w_2 \) is exposed in \( P \), proceed as in case 2a.

(a) Suppose that \( w_1a_1 \) is a \( P \)-edge and that the arc \( w_2a_2 \) is covered by (i) the \( P \)-edge \( w_2a_2 \), or (ii) by the \( P \)-path \( w_2a_2a' \). These cases are shown in Figure 3.17.

![Figure 3.17: Example diagrams for case 3a](image)

Consider the graph \( G \setminus w_2a_2 \). In case (i), there are \( k+2 \) more vertices covered by \( P' \) than by \( P \). In case (ii), this number is \( k+1 \). In both cases, \( vw_2 \) and \( w_2v \) are augmenting configurations and so there are \( k \) other vertices in (i) and \( k-1 \) in (ii) which start augmenting configurations.
in $G \setminus w_2a_2$. The remainder of case (i) is analogous to case 1b and case (ii) is analogous to case 1c, since $w_1$ and $a_1$ are not affected by the removal of the edge $w_2a_2$.

If the direction of $w_1v$ is reversed, then case (i) is the same as case 3b (i) and case (ii) is unaffected.

(b) Finally, suppose $vw_2a_2$ and $vw_1a_1$ are middles. Then (i) $a_2w_2$ is a $P$-edge, or (ii) $a_2w_2a'_2$ is a $P$-path. These cases are shown in Figure 3.18.

Figure 3.18: Example diagrams for case 3b

Consider the graph $G \setminus w_2a_2$. The number of vertices covered by $P'$ and not by $P$ is $k + 2$ in case (i) and $k + 1$ in case (ii). If we disregard the augmenting configurations beginning at $v$ and $w_2$ in case (i) and beginning at $v$ in case (ii), there remain $k$ other vertices which begin augmenting configurations in both case (i) and case (ii).

Consider case (i). The vertex $a_2$ is exposed and so it may start or end augmenting configurations. Suppose an augmenting configuration starts at $a_2$. Then $vw_2a_2$ starts this configuration in $G$. If we have an augmenting configuration $u_0\ldots u_ka_2$ of type 1, then if $u_ka_2w_2$ is a $P_2$, we have an augmenting configuration of type 3a in $G$. Otherwise, $a_2w_2v$ provides a new ending of type 1 in $G$. An augmenting configuration cannot end at $w_2$ since $w_2$ is the end vertex of the $P'$-path $w_1vw_2$ and so no other $P'$-edge can contain $w$.

For case (ii), see Lemma 3.3.1. Again, we cannot have an augmenting configuration of type 2a with $a'_2 = u_{j+1}$ and $w_2 = v_j = z$, since $w_2$ is incident with the $P'$-path $w_1vw_2$ in $G$. 
If the direction of \( w_1 w_2 \) is reversed, then case (i) is the same as case 3a (i). For case (ii), reverse the roles of \( a_2 \) and \( a'_2 \) and the case is identical.

In all cases, if there are \( k \) vertices of \( G \) exposed in \( P \), then there are \( k \) vertices which begin augmenting configurations of \( P \) in \( G \). This completes the proof of our theorem. \( \square \)

### 3.4 Packing with Directed Cycles

In this section, we consider packing with one-element families of directed cycles. We will not consider the \( \{C_2\} \)-packing problem since it is polynomial time solvable analogously to the \( \{P_1\} \)-packing problem of Theorem 3.2.1.

#### The Family \( \mathcal{G} = \{C_3\} \)

**Theorem 3.4.1.** The \( \{C_3\} \)-factor problem is \( \mathcal{NP} \)-complete.

**Proof:** This problem is in the class \( \mathcal{NP} \) since given a \( \{C_3\} \)-factor, we can check that all vertices are covered by one and only one \( C_3 \) in polynomial time. In order to prove the \( \{C_3\} \)-factor problem \( \mathcal{NP} \)-complete, we will reduce 3DM to the \( \{C_3\} \)-factor problem.

Consider the gadget of Figure 3.19 having connector vertices \( a_1, a_2, \) and \( a_3 \). We construct a graph \( G \) in the following way. Let the vertex set of \( G \) be \( W \cup X \cup Y \). For each triple of \( T = \{(t_1, t_2, t_3) \mid t_1 \in W, t_2 \in X, t_3 \in Y\} \), add a copy of the gadget to the graph by identifying \( a_1 \) with \( t_1 \), \( a_2 \) with \( t_2 \) and \( a_3 \) with \( t_3 \).

**Claim 3.4.1.** The set \( T \) has a 3-dimensional matching if and only if \( G \) has a \( \{C_3\} \)-factor.

**Proof:** Suppose \( T \) has a 3-dimensional matching. Then for each triple in the matching we have the \( \{C_3\} \)-factor of Figure 3.20.

For each triple not in the matching, we have the \( \{C_3\} \)-factor of Figure 3.21.

Clearly there is always a \( \{C_3\} \)-factor of \( G \) when there is a 3-dimensional matching of \( T \).
Suppose we have a $\{C_3\}$-factor of $G$. This part of the proof is similar to that of Theorem 2.1.1, the undirected case of this problem, with the exception that here we have connectors $a_1$, $a_2$ and $a_3$ in place of $a$, $b$ and $c$. The reader can see that just as in Theorem 2.1.1, if a connector, say $a_1$, is covered by $a_1x_2x_1$, then the only way to cover $x_3$ is to have $x_3y_3z_3$ in the $\{C_3\}$-factor and this in turn necessitates having $y_1a_2y_2$ and $z_1a_3z_2$ in the $\{C_3\}$-factor. In this case, we choose the triple $(t_1, t_2, t_3)$ of $T$ to be in the 3-dimensional matching. If $a_1$ is covered by a $C_3$ other than $x_1a_1x_2$, then in order to cover $x_1$ and $x_2$, we must choose $x_1x_2x_3$. Eventually, we find that $a_2$ and $a_3$ must
Figure 3.21: A $\{C_3\}$-factor excluding the connector vertices

also be covered by $\bar{C}_3$'s other than $y_1a_2y_2$ and $z_1a_3z_2$. In this case, we do not choose the triple $(t_1, t_2, t_3)$ of $T$ to be in the 3-dimensional matching. This produces a 3-dimensional matching of $T$. □

Thus, 3DM $\simeq \{C_3\}$-factor problem and so the $\{C_3\}$-factor problem is $NP$-complete. □

The Family $G = \{C_k\}$

**Theorem 3.4.2.** Let $k \geq 3$ be an integer. The $\{C_k\}$-factor problem is $NP$-complete.

**Proof:** As was the case for the $\{C_3\}$-factor problem, the $\{C_k\}$-factor problem is in $NP$. We will give the construction of the gadget using the case $\bar{C}_4$ as our example. The proof of this theorem uses reduction of the $k$-dimensional matching problem.

**Construction:** To construct a gadget for this problem, we begin with a directed $\bar{C}_k$ having vertices $y_1, \ldots, y_k$. To each of the vertices of this central cycle we attach $\bar{C}_k$'s which are vertex-disjoint from each other. To each of these $k \bar{C}_k$'s we attach a connector vertex $a_i$, $i = 1 \ldots k$, using two directed edges. These edges are incident with the two vertices on an edge with a vertex $y_i$, $i = 1, \ldots, k$. The addition of these connector vertices produces $k$ new $\bar{C}_k$s. An example of the gadget for $k = 4$ is given in Figure 3.22.
As in our other proofs, we construct a graph $G$ with vertex set $W_1 \cup W_2 \cup \cdots \cup W_k$. For each $k$-tuple of $T = \{(t_1, \ldots, t_k) | t_i \in W_1, \ldots, t_k \in W_k\}$ we add a copy of the gadget to $G$ by identifying connector vertex $a_1$ with vertex $t_1$, $a_2$ with $t_2$ et cetera.

Now, we claim that $T$ has a $k$-dimensional matching if and only if $G$ has a $\{C_4\}$-factor. Once again, consider the case $k = 4$. If $T$ has a 4-dimensional matching, then for a 4-tuple in the matching, the corresponding copy of the gadget in $G$ has the $\{C_4\}$-factor of Figure 3.23.

If a 4-tuple is not in the matching then we have the $\{C_4\}$-factor of Figure 3.24 of that copy of the gadget in $G$.

Thus $G$ has a $\{C_4\}$-factor.

We continue the argument as in the proof of Theorem 3.4.1 to prove that $k\text{DM} \propto \{C_k\}$-factor problem and so the $\{C_k\}$-factor problem is $\mathcal{NP}$-complete. \qed

### 3.5 Packing with Directed Stars

The $\{S_1\}$-factor problem is polynomial as shown in Theorem 3.2.1.

The $\{S_2\}$-factor problem is $\mathcal{NP}$-complete. This can be proven by reduction of 3DM. The gadget used is a modification of the gadget for the $\{P_2\}$-factor problem and is shown in Figure 3.25. The proof of $\mathcal{NP}$-completeness is similar to that of Theorem 2.3.1.
We will give a full proof of the \( \mathcal{NP} \)-completeness of the \( \{S_3\} \)-factor problem and then show how to construct a gadget which can be used to prove the \( \mathcal{NP} \)-completeness of any \( \{S_k\} \)-factor problem through reduction of the \( k + 1 \)-dimensional matching problem. The \( \mathcal{NP} \)-completeness of the \( \{S_k\} \)-factor problem will not be shown as it is similar to that of the \( \{S_3\} \)-factor problem with the obvious alterations and renaming.

The Family \( G = S_3 \)

**Theorem 3.5.1.** The \( \{S_3\} \)-factor problem is \( \mathcal{NP} \)-complete.

This problem is in the class \( \mathcal{NP} \). We will reduce \( 4DM \) to the \( \{S_3\} \)-factor problem to prove \( \mathcal{NP} \)-completeness. The gadget for this problem is given in Figure 3.26.

We will construct the graph \( G \) on vertex set \( W_1 \cup W_2 \cup W_3 \cup W_4 \). For each 4-tuple \( (t_1, t_2, t_3, t_4) \) of \( T \), add a copy of the gadget to \( G \) by identifying \( a_1 \) with \( t_1 \), \( a_2 \) with \( t_2 \), \( a_3 \) with \( t_3 \), and \( a_4 \) with \( t_4 \).

**Claim 3.5.1.** \( T \) has a 4-dimensional matching if and only if \( G \) has an \( \{S_3\} \)-factor.

**Proof:** Suppose there is a 4-dimensional matching of \( T \). Then for 4-tuples in the matching, we have the \( \{S_3\} \)-factor of Figure 3.27 for the copy of gadget associated with that 4-tuple in \( G \).
Figure 3.21: A $\overline{\{C_4\}}$-factor excluding the connector vertices

Figure 3.25: Gadget for the $\{S_2\}$-factor problem

For those 4-tuples which are not in the matching, we have the factor of Figure 3.28 of the corresponding copy of the gadget.

Suppose $G$ has an $\{S_3\}$-factor. Consider a copy of the gadget. If $a_1$ is covered by the $S_3$ centred at $x_1$ and not containing $y$, then in order to cover $y$, the $S_3$ with vertices $z_4$, $z_3$, $z_2$ and $y$ must be in the $\{S_3\}$-factor. This would produce the factor shown in Figure 3.27. In this case, we choose the 4-tuple $(t_1, t_2, t_3, t_4)$ to be in the 4-dimensional matching of $\mathcal{T}$.

If $a_1$ is not covered by the $S_3$ centred at $x_1$ and not containing $y$, then a similar argument proves that the factor of Figure 3.27 is the only one possible and we do not choose the 4-tuple $(t_1, t_2, t_3, t_4)$ to be in
the 4-dimensional matching of $T$.

This gives a 4-dimensional matching of $T$. □

Therefore, the $\{S_3\}$-factor problem is $NP$-complete. □

The Family $G = S_k$

As for the $\{C_k\}$-factor problem, we will show how to construct a gadget for any $k$ to be used in proving the $NP$-completeness of the $\{S_k\}$-factor problem. Once again, we will demonstrate this construction for the case $k = 4$.

Theorem 3.5.2. The $\{S_k\}$-factor problem is $NP$-complete.

Construction: Consider an $\tilde{S}_k$ centred at a vertex $y$ and having arcs $(y, z_i)$, $i = \{2, \ldots, k+1\}$. Attach an $\tilde{S}_k$ centred at a vertex $x_1$ to $y$ with the arc $x_1y$. Similarly, attach an $\tilde{S}_k$ centred at a vertex $x_i$, $i = \{2, \ldots, k+1\}$ to each vertex $z_i$ with the arc $z_ix_i$. Choose one of the unnamed vertices of each star centred at $x_i$ and name it $a_i$, $i = 1, \ldots, k+1$. These are the $k+1$ connector vertices of the gadget. When $k = 4$, this construction produces the gadget of Figure 3.29.

To prove Theorem 3.5.2, reduce the $k+1$-dimensional matching problem to the $\{S_k\}$-factor problem as before. The rest of the proof is left to the reader. □
CHAPTER 3. DIRECTED GRAPH PACKINGS

Figure 3.27: An \( \{S_3\}\)-factor containing the connector vertices

Figure 3.28: An \( \{S_3\}\)-factor excluding the connector vertices
Figure 3.29: Gadget for the $\{S_4\}$-factor problem
Chapter 4

Conclusion

The purpose of this thesis was to study the graph packing problem for directed graphs. Much work had been done on the undirected graph packing problem and so we felt that study of the directed cases might yield interesting results. Our work was based upon both matching theory and the work of many mathematicians on various undirected graph packing problems. As a result, the second chapter was dedicated to giving the reader an introduction to some relevant results on the undirected packing problem. In addition, we endeavoured to demystify to some extent certain areas of \( \mathcal{NP} \)-completeness theory pertinent to our work. We trust that after reading this thesis, the reader has some understanding of the realm of \( \mathcal{NP} \)-completeness and of the usefulness of this theory in studying and proving the various degrees of difficulty of problems. Our \( \mathcal{NP} \)-completeness proof technique of choice is the local replacement technique which employs the use of gadgets as demonstrated extensively in Chapter 3. Other methods of proof exist and the reader is encouraged to peruse [24].

We do not claim that Chapter 2 provides a complete survey of all work done on undirected graph packings. The graph packing problem has been approached and defined in so many different ways that a complete survey would produce many volumes of work. We do hope however, that Chapter 2 provides a good representation of the work done on the type of undirected graph packing problems which we chose to study in the directed case.

In Chapter 3, we studied directed graph packing problems for families of directed paths, directed cycles and directed stars. The focus of the chapter was the statement and proof of the main theorem of the thesis, which states that the directed \( G \)-packing problem is polynomial for the family \( G = \{\overline{P_1}, \overline{P_2}\} \).

To conclude, we provide a summary of the complexity of the problems presented in this
thesis. Recall that a polynomial problem can be solved by a polynomial-time algorithm, whether such an algorithm has been found or not is another question! \( \mathcal{NP} \)-completeness is a classification of problems of a certain degree of difficulty. Why is it interesting to know that a problem is one of these difficult types? Well, to answer that, we defer to a nicely stated explanation from [24].

Indeed, discovering that a problem is \( \mathcal{NP} \)-complete is usually just the beginning of work on the problem. . . . However, the knowledge that it is \( \mathcal{NP} \)-complete does provide valuable information about what lines of approach have the potential of being most productive. Certainly the search for an efficient, exact algorithm should be accorded low priority. It is now more appropriate to concentrate on other, less ambitious, approaches. For example, you might look for efficient algorithms that solve various special cases of the general problem. You might look for algorithms that, though not guaranteed to run quickly, seem likely to do so most of the time. . . . In short, the primary application of the theory of \( \mathcal{NP} \)-completeness is to assist algorithm designers in directing their problem-solving efforts toward those approaches that have the greatest likelihood of leading to useful algorithms.

Table 4.1: Complexity of the packing problem for families of cliques

<table>
<thead>
<tr>
<th>Family</th>
<th>Polynomial</th>
<th>( \mathcal{NP} )-Complete</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>{( K_1 )}</td>
<td>•</td>
<td></td>
<td>[20, 19, 22]</td>
</tr>
<tr>
<td>{( K_2 )}</td>
<td>•</td>
<td></td>
<td>[36]</td>
</tr>
<tr>
<td>{( \alpha \cdot K_1 \cup \beta \cdot K_2 )}</td>
<td>•</td>
<td></td>
<td>[36]</td>
</tr>
<tr>
<td>{( K_n ), ( n \geq 3 )}</td>
<td>•</td>
<td></td>
<td>[35, 36]</td>
</tr>
<tr>
<td>{( K_2, K_3 )}</td>
<td>•</td>
<td></td>
<td>[29]</td>
</tr>
<tr>
<td>( G \subseteq {K_1, K_2, K_3, \ldots } ), ( K_1 \in G ) or ( K_2 \in G )</td>
<td>•</td>
<td></td>
<td>[17, 27, 29, 36]</td>
</tr>
<tr>
<td>( G \subseteq {K_1, K_2, K_3, \ldots } ), ( K_1, K_2 \in G )</td>
<td>•</td>
<td></td>
<td>[29]</td>
</tr>
</tbody>
</table>
Table 4.2: Complexity of the packing problem for families of complete bipartite graphs

<table>
<thead>
<tr>
<th>Family</th>
<th>Polynomial</th>
<th>( \mathcal{NP} )-Complete</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>( {S_1, S_2, \ldots, S_k} )</td>
<td>\bullet</td>
<td>\bullet</td>
<td>[28, 30]</td>
</tr>
<tr>
<td>( S = {S_1, S_2, \ldots, } )</td>
<td>\bullet</td>
<td>\bullet</td>
<td>[28, 30]</td>
</tr>
<tr>
<td>( S ) not a sequential set of stars</td>
<td>\bullet</td>
<td></td>
<td>[28, 30, 36]</td>
</tr>
</tbody>
</table>

Table 4.3: Complexity of the packing problem for families of paths

<table>
<thead>
<tr>
<th>Family</th>
<th>Polynomial</th>
<th>( \mathcal{NP} )-Complete</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>( {P_k}, k \geq 2 )</td>
<td>\bullet</td>
<td>\bullet</td>
<td>[3, 28]</td>
</tr>
<tr>
<td>( {P_1, P_2} )</td>
<td>\bullet</td>
<td></td>
<td>[3, 28]</td>
</tr>
</tbody>
</table>

Table 4.4: Complexity of the packing problem for families of cycles

<table>
<thead>
<tr>
<th>Family</th>
<th>Polynomial</th>
<th>( \mathcal{NP} )-Complete</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>( {C_3, C_4, \ldots} ) (Two-factors)</td>
<td>\bullet</td>
<td>\bullet</td>
<td>[20, 22, 29, 39]</td>
</tr>
<tr>
<td>( {C_3, C_5, C_6, \ldots} )</td>
<td>\bullet</td>
<td>\bullet</td>
<td>[29]</td>
</tr>
<tr>
<td>( {C_4, C_5, \ldots} )</td>
<td>\bullet</td>
<td>\bullet</td>
<td>[29]</td>
</tr>
<tr>
<td>( {C_5, C_6, \ldots} )</td>
<td>\bullet</td>
<td>\bullet</td>
<td>[29]</td>
</tr>
<tr>
<td>All other ( G \subseteq {C_3, C_4, \ldots} )</td>
<td>\bullet</td>
<td>\bullet</td>
<td>[29]</td>
</tr>
<tr>
<td>( {C_3, C_4, C_6, C_7, \ldots} )</td>
<td>\bullet</td>
<td>\bullet</td>
<td>[18]</td>
</tr>
<tr>
<td>( {C_3, C_5, C_7, \ldots} )</td>
<td>\bullet</td>
<td>\bullet</td>
<td>[18]</td>
</tr>
<tr>
<td>( {C_4, C_6, C_7, \ldots} )</td>
<td>\bullet</td>
<td>\bullet</td>
<td>[18]</td>
</tr>
<tr>
<td>( {C_6, C_7, C_8, \ldots} )</td>
<td>\bullet</td>
<td></td>
<td>[18]</td>
</tr>
</tbody>
</table>
Table 4.5: Complexity of the packing problem for mixed families

<table>
<thead>
<tr>
<th>Family</th>
<th>Polynomial</th>
<th>$\mathcal{NP}$-Complete</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>${K_2, H}$, $H$ hypomatchable</td>
<td>$\bullet$</td>
<td></td>
<td>[17, 16, 29]</td>
</tr>
<tr>
<td>${K_2, H}$, $H$ perfectly matchable</td>
<td>$\bullet$</td>
<td></td>
<td>[41, 40]</td>
</tr>
<tr>
<td>${K_2, H}$, $H$ a propeller</td>
<td>$\bullet$</td>
<td></td>
<td>[41, 40]</td>
</tr>
<tr>
<td>${K_2, H}$, $H \neq$ propeller, hypomatchable, perfectly matchable</td>
<td>$\bullet$</td>
<td></td>
<td>[41, 40]</td>
</tr>
<tr>
<td>${K_2, F_1, \ldots, F_k}$, $F_i$ hypomatchable</td>
<td>$\bullet$</td>
<td></td>
<td>[29]</td>
</tr>
</tbody>
</table>

Table 4.6: Complexity of the packing problem in the directed case

<table>
<thead>
<tr>
<th>Family</th>
<th>Polynomial</th>
<th>$\mathcal{NP}$-Complete</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>${\overrightarrow{P}_1}$</td>
<td>$\bullet$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>${\overrightarrow{P}_k}$, $k \geq 2$</td>
<td>$\bullet$</td>
<td>$\bullet$</td>
<td></td>
</tr>
<tr>
<td>${\overrightarrow{P}_1, \overrightarrow{P}_2}$</td>
<td>$\bullet$</td>
<td></td>
<td>[12]</td>
</tr>
<tr>
<td>${\overrightarrow{P}_1, \overrightarrow{P}_j}$, for $j$ odd, $j \geq 3$</td>
<td>$\bullet$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>${\overrightarrow{P}_1, \overrightarrow{P}_j}$, for $j$ even, $j \geq 4$</td>
<td>$\bullet$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\mathcal{G} \subseteq {\overrightarrow{P}_k, \ldots, \overrightarrow{P}<em>l, \ldots}$, $2k-2 \geq l \geq k$, $k \geq 2$, $\overrightarrow{P}</em>{l+1} \notin \mathcal{G}$</td>
<td>$\bullet$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>${\overrightarrow{C}_k}$, $k \geq 3$</td>
<td>$\bullet$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>${\overrightarrow{S}_k}$, $k \geq 2$</td>
<td>$\bullet$</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Much remains to be studied in the area of directed graph packings. For example, packing problems with families of bipartite or complete bipartite directed graphs, families of two or more directed cycles or directed stars have not been considered. We hope that the work in this thesis provides a helpful beginning to the study of the directed graph packing problem.
Bibliography


BIBLIOGRAPHY


