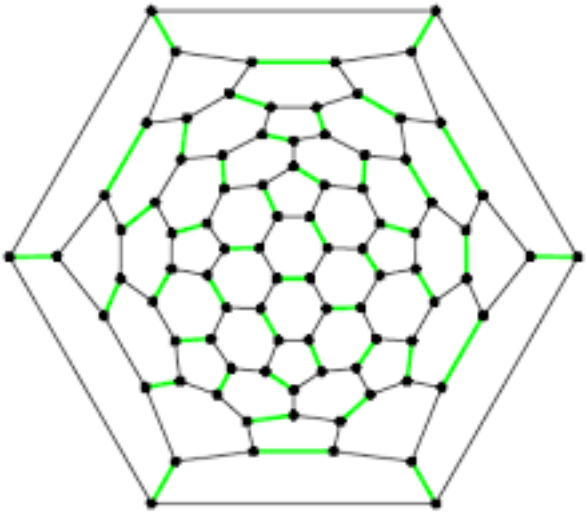
Bipartite Matching : Framework and Solutions

A path to the Hungarian Algorithm and beyond

Giancola Simone Maria[1†]

[†]Bocconi University, Milan

September 13, 2022



[1]simonegiancola09@gmail.com

Abstract

In a scenario of tasks and agents that could perform them at a specified cost, the Assignment Problem aims to find the best pairs that attain an optimal cost. Such a question is very common across industries, especially with scarcity of resources, and an efficient solution is of paramount importance. It is also theoretically linked with its unweighted version, commonly referred to as Bipartite Matching, thanks to which the conditions for a solution can be identified.
First, the problem is presented from a mathematical perspective. In the second chapter, properties from the graph theory and linear algebra spectrum are proposed, highlighting a meeting point of the two subjects. Lastly, a solution through the Hungarian Algorithm by Kuhn [1] is analyzed and justified using prior results. The whole reasoning process is backed by formal justification, with proofs and a clear set of defined objects. Two additional chapters provide extensions in the direction of different optimization processes and magnitude of the solutions in the problem space. The former is a generalization of the proposed framework. The latter is an overview on the number of such feasible solutions in the unweighted case for a quite general type of graph.

Contents

List of Figures

List of Tables

List of Algorithms

List of Theorems

List of Symbols

$\mathfrak{P}$	bipartite matching problems
$\mathscr{A}$	agents set
$\mathscr{T}$	tasks set
C	cost function or matrix
Π	space of bijections
n	size of problem
$\mathfrak{C}$	combinatorial optimization problems
Φ	feasibility check function
$\mathscr{S}$	feasible arrangements set
M	minimum or maximum function
$T(n)$	computational time in terms of size
$\mathfrak{M}$	most general space of matrices
$\mathscr{X}$	permutation matrices
$\mathfrak{G}$	space of undirected graphs
$\mathscr{B}$	space of bipartite graphs
$\mathscr{V}, \mathscr{W}$	sets of vertices
$\mathscr{E}$	set of edges
$\mathscr{M}$	matching set of edges
$\mathscr{K}$	complete bipartite graphs
$\nu(G)$	maximum cardinality perfect matching size
$\delta(\cdot)$	neighbors set
A	incidence matrix graph
E	number of edges in a complete bipartite graph
$\mathscr{X}$	doubly stochastic matrices space
$\mathcal{H}$	hyperplane
$\mathbf{C}_S$	convex hull of a set
$\mathcal{Z}$	polyhedron
$\mathcal{P}$	polytope
$\mathscr{Z}$	space of linearly constrained regions
$\mathcal{A}$	totally unimodular matrices space
$\mathscr{C}$	cover set of vertices
$\tau(G)$	minimum size vertex cover
$\mathfrak{D}$	vertex cover problems
Δ	symmetric set difference
$\mathcal{C}$	stable set
$\alpha(G)$	maximum size stable set
$\mathcal{M}$	edge cover

$\rho(G)$	minimum size edge cover
$\kappa(\cdot)$	fractional cover function
$\mathcal{R}_{\mathscr{M}}$	alternating path or route
$\mathfrak{Q}$	directed graphs space
$\mathscr{Q}$	directed bipartite graphs space
$\mathcal{L}_{\mathscr{M}}$	labelling set of vertices
$\omega(\cdot)$	slack weight function
D_{start}	starting dual
B_{ω}	complementary slackness subgraph
$\preceq$	total order relation
$\mathcal{S}$	totally ordered commutative semigroup
$\zeta, \zeta(T)$	admissible transformation
S_n	symmetric group
$per(\cdot)$	permanent of a matrix
$\mathcal{C}^{\circlearrowleft}$	cycle cover
$det(\cdot)$	determinant of a matrix
$\mathcal{T}$	spanning tree
L	Laplacian matrix
$Pf(\cdot)$	pfaffian
$\mathscr{F}$	faces of a planar graph
$\mathcal{G}$	dual planar graph

Chapter 1

Introduction

The best theory is inspired by practice. The best practice is inspired by theory.

Donald Knuth, 1974 Turing Award

We begin the analysis with a motivating real life example.

Example 1.1 (Resource allocation problem)**.** An employer is looking simultaneously for a President, a CEO and a COO. Having found 3 potential employees, the aim is minimizing the total cost. The cost allocation options can be seen through a matrix:

	President	**CEO**	**CFO**
George	40	2	45
Paul	1	20	30
Kristine	50	62	3

Table 1.1: Problem representation

A quick look is sufficient to notice that the optimal allocation has cost $C = 6$ where:

- Paul is hired as President
- George is hired as CEO
- Kristine is hired as CFO

While the above scenario returns an easy configuration, this is not always the case.
In a general setting, this is referred to as the famous balanced assignment problem, or LSAP[1] in some references such as [2]. A characterization is proposed below.

Definition 1.2 (Balanced Assignment Problem $\mathfrak{P}$)**.** Given a set of agents $\mathscr{A}$ a set of tasks $\mathscr{T}$ such that[2] $|\mathscr{A}| = |\mathscr{T}| < \infty$ and a cost function $C : \mathscr{A} \times \mathscr{T} \to \mathbb{Z}_+$ assign one task per agent and one agent per task such that the total cost is minimized. Namely find:

$$\pi^* \in \Pi \coloneqq \{self\ bijections\} \quad : \quad \pi^* : \mathscr{A} \to \mathscr{T} \ : \ \pi^* = argmin_{\pi \in \Pi} \left\{ \sum_{a \in \mathscr{A}} C(a, \pi(a)) \right\} \tag{1.0.1}$$

An instance of $\mathfrak{P}$ is denoted as P.

[1] Linear Sum Assignment Problem

[2] Later in the text it will be formally shown why any instance reduces to the equal size case.

The interest in studying solving methods for $P \in \mathfrak{P}$ arises from its nature. Indeed, it falls into the the broader set of combinatorial optimization problems.

Definition 1.3 (Combinatorial Optimization Problem $\mathfrak{C}$)**.** A combinatorial optimization problem is a quadruplet $\left(\mathscr{P}, \Phi, C, M\right)$ where:

- $\mathscr{P}$ is a family of problem instances
- Φ is a function $\Phi : \mathscr{P} \to \mathscr{S}$ which returns **feasible** solutions
  - Where, more specifically $|\mathscr{S}| < \infty$
- C is a function $C : \mathscr{S} \to \mathbb{Z}_+$ returning the cost of a feasible solution
- M is a function $M : \mathscr{S} \to \mathscr{S}$ either min or max

Where for $P \in \mathfrak{P}$ the task is to find:

$$C(S^*) = M\left[C(H) \mid H \in \Phi(\mathscr{P}_P)\right] \tag{1.0.2}$$

In other words, the objective is to find the feasible solution S^* of the problem instance P that attains minimum or maximum cost.
Intuitively, for each combinatorial optimization problem there is a paired **Decision Problem** which explores the existance of a feasible solution. The possibility of having an instance P with no solutions will be dealt with in the following pages.

Observation 1.4. It is easy to see that $\mathfrak{P}$ can be represented as in $\mathfrak{C}$. We have a cost function C to minimize in a finite landscape $\mathscr{P}$"="$\{\mathscr{A}, \mathscr{T}\}$ of configurations combined in a feasible way forming:

$$\mathscr{S} \subseteq (\mathscr{A} \times \mathscr{T})^n \implies |\mathscr{S}| < (|\mathscr{A}| \times |\mathscr{T}|)^n = n^{2n} < \infty$$

1.1 A first Strategy

While someone might claim that there are easy problems with a representation $\mathfrak{C}$ or easy instances, this is not a general rule. For the Assignment problem the aim is to find a method to solve any $P \in \mathfrak{P}$.

Fact 1.5 (First Strategy: Enumeration)**.** Not to be considered as a real algorithm. We take into account just plain enumeration and evaluation of feasible options.

Assuming that the cost evaluation takes time $O(1)$ for simplicity, attemping to assign the agents $a \in \mathscr{A}$ to tasks $t \in \mathscr{T}$ without repetition gives the following recursion:

- For the first agent a_1 there are $|\mathscr{T}| = n$ choices
- For the second agent a_2 there are $n - 1$ task choices as one was taken
- ...
- For the n^{th} agent a_n there is only one choice as all were taken

This reduces to having $|\mathscr{S}| = n!$ possible arrangements, which would require as a computational time with respect to size $T(n) \in O(n!)$ operations.

Hence, $\mathfrak{P}$ is a combinatorially exploding in size problem at first sight. While there might be easy instances, not all of them are straightforward.

The $n!$ result is actually the effect of constraining π to be part of the space of bijections Π. It can be noticed though that this is not advantageous for enumeration.

Observation 1.6 (Constraints effect on size)**.** *Using Stirling's approximation* $n! \sim \sqrt{2\pi n}\left(\frac{n}{e}\right)^n$ *we also have that the ratio of feasible solutions with respect to the ratio of the possible solutions is around:*

$$\frac{|\mathscr{S}|}{|(\mathscr{A} \times \mathscr{T}|)^n} = \frac{n!}{n^{2n}} \simeq \frac{\sqrt{2\pi n}(\frac{n}{e})^n}{n^{2n}} = \sqrt{2\pi n}\frac{1}{(en)^n} \overset{n\to\infty}{\to} 0 \tag{1.1.1}$$

And yet this decrease is not enough to render the set of feasible solutions $\mathscr{S}$ *tractable. This means that even by setting rules and strongly restricting the possible configurations, time efficiency is not guaranteed.*

Example 1.7 (An enumeration attempt)**.** A set of $n = 30$ tasks and agents with a clear cost function is given. Assume we have at disposal a computer that does $2.59 \cdot 10^9$ operations per second (just to simplify things). Usually, we are around hundreds of millions per seconds, so consider it to be **very** powerful.
Let $n = 30$ tasks, then:

$$n! = n(n-1)(n-2)\dots(n-(n-2))(n-(n-1)) = \prod_{i=0}^{n}(n-i) = 30! \approx 2.65 \cdot 10^{32} \tag{1.1.2}$$

This is clearly a large number of possible options to evaluate. Still, someone might say that a good procedure would be enumerating all of them and evaluating the cost for each. Assuming that the calculation is done efficiently in negligible time, we would still have 30! enumerations to perform. The time[3] required for this strategy is:

$$\frac{30!}{2.65 \cdot 10^9} sec = 10^{23} sec \equiv 31709791983764584\ years \tag{1.1.3}$$

$$\approx 3 \cdot 10^{16}\ years \approx 2 \times 10^6 \text{ stories of the universe} \tag{1.1.4}$$

Clearly this is not the most efficient way to tackle the problem!

1.2 Representations of the Assignment problem

It is often the case that problems such as that of Definition 1.2 are solved through specific mathematical representations that give rise to "nice" properties. In this section, the main ones will be outlined.

Definition 1.8 (Permutation π)**.** A function returning a permutation of indices.

$$\pi : \{1, \dots, n\} \to \{1, \dots, n\},\ \pi \in \Pi$$

Clearly a permutation is a self bijection and viceversa.

Definition 1.9 (Matrix Form of $\mathfrak{P}$)**.** Given $P \in \mathfrak{P}$, the function $\pi(a)$ of Definition 1.2 can be interpreted as a permutation. Imagine assigning a number $i \in \{1, \dots, n\} \forall a_i \in \mathscr{A}$ and doing the same for elements of $\mathscr{T}$. Clearly, for each item there is a unique corresponding number. Moreover, the cost function C can be interpreted as a matrix where $C_{ij} = C(a_i, t_j) \in \mathfrak{M}_{n,n}$. Considering a permutation as in Definition 1.8, where $\pi : \mathscr{A} \to \mathscr{T}$ with the elements indexed by integers and

$$\pi(a_i) = t_j\ : \forall i, j\ \ a_i \in \mathscr{A},\ t_j \in \mathscr{T}$$

[3]Assuming the universe is about 13.8 Billion years old, first google suggestion

It is possible to build a matrix:

$$X = \begin{bmatrix} e_{\pi(a_1)} \\ \dots \\ e_{\pi(a_n)} \end{bmatrix} \in \mathcal{X} \subseteq \mathfrak{M}_{n,n} \tag{1.2.1}$$

Where e_i is the elementary unit vector with 1 in the i^{th} entry, and $\mathcal{X}$ is the space of permutation matrices. This space is naively composed of matrices where there are n non zero entries made of 1s, placed such that there is only one per row and one per column[4]. Having this permutation matrix model, problem $\mathfrak{P}$ can be interpreted as finding the optimal $X \in \mathcal{X}$ for the cost matrix C such that the trace of the result is minimized. Namely, find:

$$X^* \in \mathcal{X} \ : \ X^* = \underset{X \in \mathcal{X}}{argmin} \left\{ trace(XC) \right\} \tag{1.2.2}$$

Where the trace is used to easily select one element per row and one per column

Definition 1.10 (Integer Programming form of $\mathfrak{P}$)**.** From definition 1.9 it is possible to formulate the problem as a linear $0, 1$ programming instance such that:

$$find \ \ X^* \in \mathcal{X} \ \ : \ \ X^* = \underset{X \in \mathcal{X}}{argmin} \left\{ \sum_{i=1}^{n} \sum_{j=1}^{n} C_{ij} x_{ij} \right\} \tag{1.2.3}$$

Where we define X as a permutation matrix such that:

$$X \coloneqq \begin{cases} x_{ij} = 1 \ \ if \ \ \pi(a_i) = t_j \ \ a_i \in \mathscr{A}, t_j \in \mathscr{T} \\ x_{ij} = 0 \ \ otherwise \end{cases} \tag{1.2.4}$$

Subject to the constraints:

$$\begin{cases} \sum_{i=1}^{n} x_{ij} = 1 \\ \sum_{j=1}^{n} x_{ij} = 1 \end{cases} \tag{1.2.5}$$

It is also possible to view $\mathfrak{P}$ as a graph problem.

Definition 1.11 (Bipartite Graphs $\mathscr{B}$)**.** Assuming $\mathfrak{G}$ is the space of undirected graphs, Bipartite Graphs in $\mathscr{B}$ present two disjoint sets of vertices that do not have any inner edge.

$$\mathscr{B} \coloneqq \left\{ B \in \mathfrak{G} \mid B = \{(\mathscr{V} \cup \mathscr{W}), \mathscr{E}\} \ : \ \nexists e \in \mathscr{E} \ e = (v_1, v_2) \ e = (w_1, w_2) \ v_1, v_2 \in \mathscr{V}, \ w_1, w_2 \in \mathscr{W} \right\} \tag{1.2.6}$$

Definition 1.12 (Complete Bipartite Graphs $\mathscr{K}_{n,m}$)**.** Complete Bipartite graphs are Bipartite graphs with all the possible edges connecting the two disjoint sets. In standard notation, n is the size of the first set, and m is the size of the second set.

$$\mathscr{K}_{n,m} \coloneqq \left\{ K \in \mathscr{B} \mid K = \{(\mathscr{V} \cup \mathscr{W}), \mathscr{E}\} \ : \ \forall v \in \mathscr{V}, \forall w \in \mathscr{W} \exists e \in \mathscr{E} \ : \ e = (v, w) \ |\mathscr{V}| = n, |\mathscr{W}| = m \right\} \tag{1.2.7}$$

[4]Notice that the number of such matrices is $|\mathcal{X}| = n!$ as in all the possible configurations of Fact 1.5. Where $|\cdot|$ means *number of distinct* and not *dimension*

Definition 1.13 (Matching $\mathscr{M}$)**.** Given $G = (\mathscr{V}, \mathscr{E}) \in \mathfrak{G}$ a matching is a collection of edges such that all the vertices are reached at most once:

$$\mathscr{M} \subseteq \mathscr{E} \ : \ \forall v \in \mathscr{V} \exists!(or\ less)\ e \in \mathscr{M},\ v \in e \tag{1.2.8}$$

A vertex $v \in \mathscr{V}$ is said to be **exposed** if no edge in $\mathscr{M}$ is incident to it.
A matching has **maximum cardinality** if it contains the maximum possible number of edges in $\mathscr{E}$. Its size is denoted as $\nu(G)$
A matching is **perfect** if it assigns an edge to each vertex in $\mathscr{V}$. It is trivially also of maximum cardinality, and no vertex in G is exposed.

Definition 1.14 (Graph form of $\mathfrak{P}$)**.** Given a Bipartite graph $B \in \mathscr{B}$ where $B = \{(\mathscr{A} \cup \mathscr{T}), C\}$ find the minimum weight perfect matching.
Following the notation of vertices and edges it is possible to infer that:

- We require $B \in \mathscr{B}$ (bipartite) as we are interested in assigning agents $a \in \mathscr{A}$ to tasks $t \in \mathscr{T}$ and not within the two.
- We wish to find a matching of maximum cardinality as among the possible connections from Definition 1.2 we have that a feasible solution of P finds a one to one agent-task assignment F
- The minimum weight feature is equivalent to minimum cost as C is the set of edges but also the possible cost pairings.

Definition 1.15 (Neighbors set generator $\delta(\cdot)$)**.** Given a graph $G = \{\mathscr{V}, \mathscr{E}\} \in \mathfrak{G}$ for each vertex we define its neighbors set generator as:

$$\delta(\cdot) \ : \ \forall v \in \mathscr{V}\ \delta(v) = \{e_v\}\ : e_v = (v, v') \text{ for some } v' \in \mathscr{V} \tag{1.2.9}$$

It is a function that given a vertex returns its neighbors.

Definition 1.16 (Incidence Matrices A)**.** An incidence matrix is a representation of the connections inside a graph, for each vertex in the rows a 1 is placed whenever it is connected with another through the corresponding edge in the columns.
Thus, for a graph $G = \{\mathscr{V}, \mathscr{E}\} \in \mathfrak{G}$ the incidence matrix $A \in \mathfrak{M}_{|\mathscr{V}|,|\mathscr{E}|}$ is:

$$A \coloneqq \begin{cases} a_{ij} = 1 & if\ e_j \in \mathscr{E} \ : \ e_j = (v_i, v), v_i, v \in \mathscr{V} \\ a_{ij} = 0 & otherwise \end{cases} \tag{1.2.10}$$

The graph perspective, together with the just introduced objects, allows for a final formulation of the assignment problem.

Definition 1.17 (Graph inspired matrix form of $\mathfrak{P}$)**.** For a given Bipartite graph $B \in \mathscr{B}$ where $B = \{(\mathscr{V} \cup \mathscr{W}), \mathscr{E}\}$ solve the following integer program:

$$\begin{cases} x^* = \underset{x \in \mathbb{R}^{|\mathscr{E}|}}{argmin} \left\{ \sum_{e=(v_i,w_j) \in \mathscr{E}} x_e C_{ij} \right\} \\ \sum_{e \in \delta(v)} x_e = 1 \forall e \\ x_e \in \{0,1\} \forall e \end{cases} \tag{1.2.11}$$

Where x^* is a vector with $\{0,1\}$ entries and $x^*_e = 1 \iff e \in \mathscr{M} \subseteq \mathscr{E}$ is selected in the perfect matching. In matrix form, it is possible to recognize the incidence matrix A in the second constraint:

$$\begin{cases} x^* = \underset{x \in \mathbb{R}^{|\mathscr{E}|}}{argmin} \left\{ x \cdot \mathbf{c} \right\} \\ Ax = \mathbf{1} \\ x_e \in \{0,1\} \forall e \end{cases} \tag{1.2.12}$$

Where $\mathbf{c}$ is a cost vector identifying the cost of each edge, and $\mathbf{1} = [1, \dots, 1]^T \in \mathbb{R}^n$.
The second constraint ensures that for each vertex only one edge is chosen.
The third constraint ensures that for each edge it is either included or not in the matching $\mathscr{M}$.

The proposed frameworks can generalize any instance of $\mathfrak{P}$, provided that some assumptions are made.

Assumption 1.18 (Vertices of the Graph)**.** Throughout the document, to simplify results the graphs do not present isolated vertices with no incident edges.

Assumption 1.19 (Size of the sets)**.** We will only consider the *balanced* case, implied by the condition $|\mathscr{A}| = |\mathscr{T}|$. Throughout the document it will be assumed that $|\mathscr{A}| = |\mathscr{T}| = n$

Observation 1.20 (About Assumption 1.19)**.** *It is possible the equal size requirement with the following two arguments:*

- *If the Cost function is not defined for some pair $\nexists C(a_i, t_j)$ we have a Bipartite graph but not a complete Bipartite graph. However, we can assign infinite cost to the missing edges:*
$$C(a_i, t_j) = \infty$$
- *If the number of agents and tasks is different $|\mathscr{A}| \neq |\mathscr{T}|$, take the least one of the two and adjust the formulation adding infinite costs for any pair involving it. If for example the set of tasks is bigger, we enlarge agents adding dummy elements a^* where:*
$$C(a^*, t_j) = \infty \forall t_j \in \mathscr{T}$$

Thanks to these two adjustments we have that:

$$B \in \mathscr{B} \rightsquigarrow B \in \mathscr{K}_{n,m} \rightsquigarrow B \in \mathscr{K}_{n,n}$$

The last form of the problem presents nicer properties, which will be outlined in the next section. Given that throughout the document it will be assumed that the graph is *bipartitely* full a notation assumption on the number of edges is added.

Assumption 1.21 (Number of edges E)**.** Given a complete bipartite graph $\mathscr{K}_{n,n}$ the number of edges is denoted as $E = n^2$. The result number is obtained with a simple combinatoric argument.

Example 1.1 can be viewed in the three proposed perspectives.

Example 1.22 (Resource Allocation mathematical forms)**.** The Cost matrix C, common to all problems, and is built assuming rows are agents and columns are tasks as:

$$C = \begin{bmatrix} 40 & 2 & 45 \\ 1 & 20 & 30 \\ 50 & 62 & 3 \end{bmatrix}$$

In matrix form from Definition 1.9:

$$X^* = \underset{X \in \mathcal{X}}{argmin} \left\{ trace(XC) \right\} \implies X^* = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

In integer programming form

$$X^* = \underset{X \in \mathcal{X}}{argmin} \left\{ \sum_{i=1}^{n} \sum_{j=1}^{n} C_{ij} x_{ij} \right\} \; s.t. \; \begin{cases} \sum_{i=1}^{n} x_{ij} = 1 \\ \sum_{j=1}^{n} x_{ij} = 1 \\ x_{ij} = 1 & if \; \pi(a_i) = t_j \;\; a_i \in \mathscr{A}, t_j \in \mathscr{T} \\ x_{ij} = 0 & otherwise \end{cases}$$

$$\implies X^* = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

As a Bipartite Matching Problem, it can be represented as shown in Figure 1.1

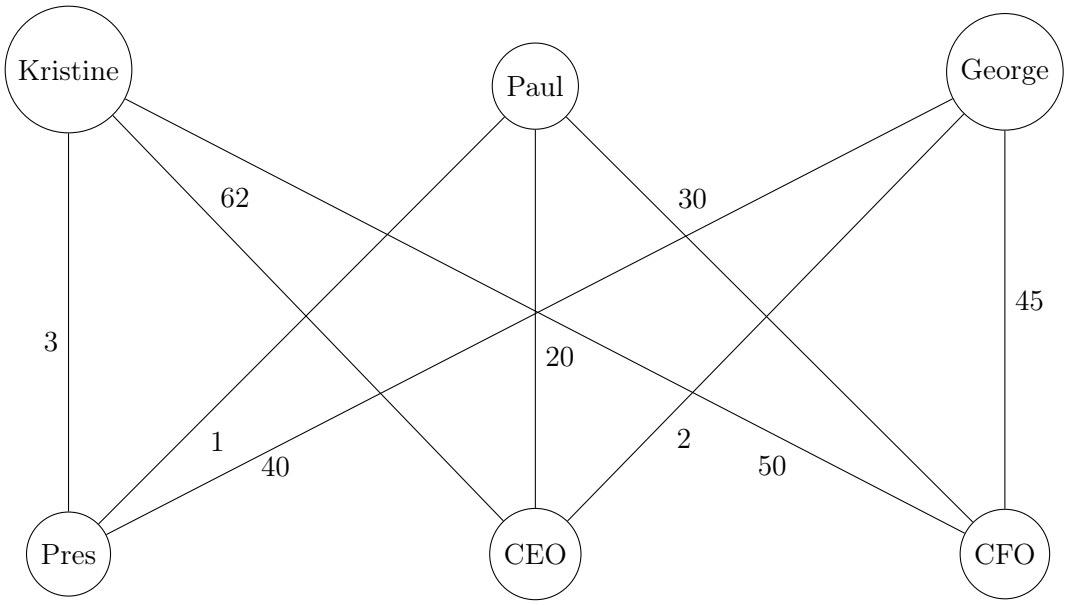


Figure 1.1: Resource Allocation Bipartite Graph

Where the minimum weight matching of maximal cardinality is $\mathscr{M}^*$, shown in Figure 1.2. Lastly, in a graph inspired matrix form as in Definition 1.17 we introduce the incidence

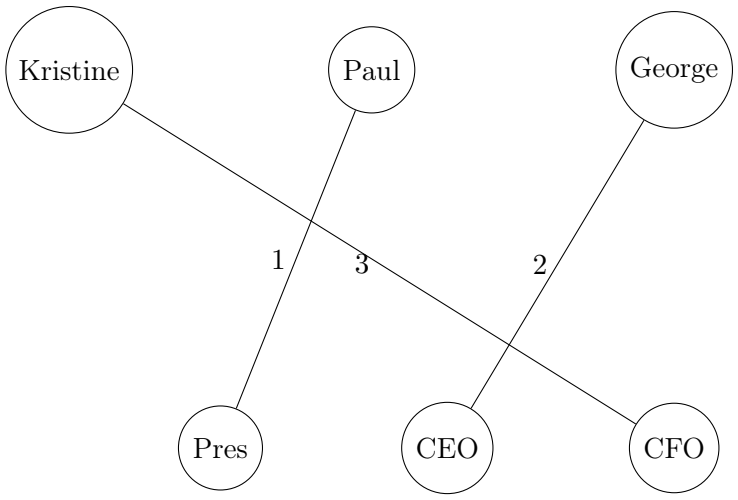


Figure 1.2: Resource Allocation solution

matrix A and the cost vector $\mathbf{c}$. To do so, a reference order is chosen, such as reading

Table 1.1 in the classical way: left right, up down. Given this convention, the following objects arise:

$$A = \begin{array}{c} \begin{matrix} G,Pr & G,C & G,CF & P,Pr & P,C & P,CF & K,Pr & K,C & K,CF \end{matrix} \\ \begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \end{pmatrix} \end{array} \begin{matrix} Geo \\ Pau \\ Kris \\ Pres \\ CEO \\ CFO \end{matrix} \tag{1.2.13}$$

$$\mathbf{c} = \begin{bmatrix} 40 \\ 2 \\ 45 \\ 1 \\ 20 \\ 30 \\ 50 \\ 62 \\ 3 \end{bmatrix} \tag{1.2.14}$$

And the optimization problem becomes:

$$\begin{cases} x^* = \underset{x \in \mathbb{R}^{|\mathscr{E}|}}{argmin} \Big\{ x \cdot \mathbf{c} \Big\} \\ Ax = \mathbf{1} \\ x_e \in \{0,1\} \end{cases} \tag{1.2.15}$$

With solution:

$$\begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \tag{1.2.16}$$

Chapter 2

Properties of $\mathfrak{P}$

Perhaps even more than to the interaction between mankind and nature, graph theory is based on the interaction of human beings with each other.

Dénes Kőnig

The advantage of using these three representations is both in terms of visualization of operations made on an instance of $\mathfrak{P}$ and to exploit features of the object.

2.1 Linear Algebra and Graph Theory facts

Some questions concerning the problem could be: :

- *Ignoring costs, what are the properties of an instance of $\mathfrak{P}$ with a maximum cardinality matching?*
- *What are the links between the graph theoretic formulation and the integer programming formulation?*
- *Is there any theoretical framework that can govern such complex problem in a controlled manner?*

Given the double nature of $\mathfrak{P}$ some facts will be outlined from a matrix perspective and others will be outlined from a graph theoretical perspective.

Definition 2.1 (Doubly Stochastic Matrices $\mathscr{X}$)**.** Doubly stochastic matrices have non negative entries and unitary row column sum.

$$\mathscr{X} \coloneqq \left\{ X \in \mathfrak{M}_{n,n} \mid x_{ij} \geq 0,\ \sum_i x_{ij} = 1,\ \sum_j x_{ij} = 1 \right\} \tag{2.1.1}$$

Definition 2.2 (Linear Programming relaxation of $\mathfrak{P}$)**.** From definition 2.1 it is possible to relax the problem as:

$$find\ \ X^* \in \mathscr{X} \quad : \quad X^* = argmin_{X \in \mathscr{X}} \left\{ \sum_{i=1}^{n} \sum_{j=1}^{n} C_{ij} x_{ij} \right\} \tag{2.1.2}$$

Subject to the constraints:

$$\begin{cases} \sum_{i=1}^{n} x_{ij} = 1 \\ \sum_{j=1}^{n} x_{ij} = 1 \\ x_{ij} \geq 0 \end{cases} \tag{2.1.3}$$

Similarly the other integer program seen in Definition 1.17 will be relaxed as:

$$\begin{cases} x^* = argmin_{x \in \mathbb{R}^{|\mathscr{E}|}} \Big\{ x \cdot \mathbf{c} \Big\} \\ Ax \leq \mathbf{1} \\ x_e \geq 0 \forall e \end{cases} \tag{2.1.4}$$

It might seem that this relaxation is useless. However, in the next steps will be clarified why this is not the case. Introducing inequality constraints, a more formal geometric framework allows the exploitance of more general results.
As a first step it can be noticed that:

Observation 2.3 (Relaxed solutions dominate Problem solutions)**.** *For a given linear relaxation of $\mathfrak{P}$, since the space of possible arrangements includes the original space of feasible solutions (e.g. $\mathcal{X} \subset \mathscr{X}$), the solution of the relaxed form will be such that the cost is at worst equal to the original optimal.*

$$\forall P \in \mathfrak{P}\ C(S^*_{rel}) \leq C(S^*) \tag{2.1.5}$$

A very important object in linear optimization is the polytope. Following the approach outlined in [3] the concept and main properties will be outlined, avoding the common appropriation of the e to different items in geometry.

Definition 2.4 (Hyperplane $\mathcal{H}_{a,c}$)**.** A hyperplane $\mathcal{H}_{a,c}$ or simply $\mathcal{H}$ in an affine space $\mathbb{R}^n$, $n \in \mathbb{N}$ is a "flat object" of dimension $n-1$ which is the inner product with an element $a \in \mathbb{R}^n$ to which the "intercept" $b \in \mathbb{R}$ is added.

$$\mathcal{H}_{a,c} = \Big\{ x \in \mathbb{R}^n \ : \ \langle x, a \rangle + b = 0 \Big\} \tag{2.1.6}$$

Definition 2.5 (Half-spaces of a point $\mathcal{H}^{\pm}$)**.** Given a hyperplane in an affine space two half spaces are created, as the object splits it into two. We denote them with signs plus and minus.

$$\mathcal{H}^{+}_{a,c} \coloneqq \Big\{ x \in \mathbb{R}^n \ : \ \langle x, a \rangle + c \geq 0 \Big\} \tag{2.1.7}$$

$$\mathcal{H}^{-}_{a,c} \coloneqq \Big\{ x \in \mathbb{R}^n \ : \ \langle x, a \rangle + c < 0 \Big\} \tag{2.1.8}$$

Definition 2.6 (Convex Hull of a Set $\mathbf{C}_S$)**.** Given a set $S \subset \mathbb{R}^n$ its convex hull is the intersection of all convex supersets of S.

This definition is cumbersome. However, given a finite number of constraints as in our problem for the case of equation 1.2.5, a convex hull ends up being the convex combination of the intersection of specific points.

Theorem 2.7 (Convex Hull of a set is a convex combination)**.** *Given a finite dimensional set $S \subset \mathbb{R}^n$ its convex hull is made of any element resulting from the convex 1-sum combination of n points belonging to S*

$$\boldsymbol{C}_S = \Big\{ \sum_{j=1}^{n} \lambda_j s_j \ : \ \forall j\ \lambda_j \geq 0\ \sum_{j=1}^{n} \lambda_j = 1, \ \ s_j \in S \Big\} \tag{2.1.9}$$

Proof. Denote the convex combination of n points as R. To show equality it suffices to show that $\mathbf{C}_S \supseteq R$ and $\mathbf{C}_S \subseteq R$.
$\supseteq$ **Direction**.
A convex superset $S' : S' \supseteq S$ trivially contains convex combinations of points inside S.
$\subseteq$ **Direction**.
It trivially holds also that $R \supseteq S$, as it is possible to assign weight $\lambda_j = 1$ for each $s_j \in S$ and null weight to the other elements, thus obtaining at least all the elements inside S. The missing requirement is proving that R is convex indeed. For this purpose two elements in R are considered:

$$r^{(1)} = \sum_{j=1}^{n} \lambda_j r_j \qquad\qquad r^{(2)} = \sum_{j=1}^{n} \nu_j r_j \tag{2.1.10}$$

Then, considering a convex combination of them with coefficient $\mu \in [0, 1]$, and applying basic algebra:

$$\begin{aligned} r =& \mu r^{(1)} + (1-\mu) r^{(2)} && \text{by definition} && (2.1.11)\\ =& \sum_{j=1}^{n} \mu\lambda_j r_j + (1-\mu)\nu_j r_j && \text{where coefficients are positive } \forall j && (2.1.12)\\ =& \sum_{j=1}^{n} [\mu\lambda_j + (1-\mu)\nu_j] r_j && \text{collecting the coefficients} && (2.1.13)\\ =& \sum_{j=1}^{n} \mu_j r_j && \text{in short form} && (2.1.14) \end{aligned}$$

Inspecting the sum of the chosen coefficients:

$$\sum_{j=1}^{n} \mu\lambda_j + (1-\mu)\nu_j = \mu \sum_{j=1}^{n} \lambda_j + (1-\mu) \sum_{j=1}^{n} \nu_j = \mu + 1 - \mu = 1 \tag{2.1.15}$$

Thus:

$$r = \mu r^{(1)} + (1-\mu) r^{(2)} = \sum_{j=1}^{n} \mu_j r_j \tag{2.1.16}$$

$$where \sum_{j=1}^{n} \mu_j = 1 \tag{2.1.17}$$

$$\implies r \in R \implies R\, convex \implies R \subseteq \mathbf{C}_S \tag{2.1.18}$$

As a consequence of bidirectional inclusion, the two sets are equivalent. □

While this result already reduces the amount of elements to combine from potentially infinite to a finite number n, it is not yet clear which elements should be taken in consideration to obtain the Convex Hull of a set. The next theorem provides more information about a potential characterization, which will also be crucial for the development of further properties.

Definition 2.8 (Polyhedron $\mathcal{Z}_{A,b}$)**.** In this document, we refer to a polyhedron as the set of solutions to a linear inequality. Given a matrix $A \in \mathfrak{M}_{n,E}$ and a vector $b \in \mathbb{R}^E$:

$$\mathcal{Z}_{A,b} \coloneqq \left\{ x \in \mathbb{R}^E \ : \ Ax \leq b \right\} \in \mathscr{Z} \tag{2.1.19}$$

Where $\mathscr{Z}$ is the space of polyhedrons, or linearly constrained regions.

Definition 2.9 (Polytope $\mathcal{P}_{A,b}$)**.** A polytope is a bounded polyhedron.

$$\mathcal{P}_{A,b} \in \mathscr{Z} \ : \ \mathcal{P}_{A,b} \textit{ bounded} \tag{2.1.20}$$

If such an object arises from the convex hull of a finite set of points $s \in S \subset \mathbb{R}^E$ it is referred to as **Convex Polytope**.

$$\mathcal{P}_S \coloneqq \mathbf{C}_{S'} \tag{2.1.21}$$

Here, it is not clear whether $S \equiv S'$ or not.

Definition 2.10 (Vertices of a Convex Polytope $Ext(\mathcal{P}_S)$)**.** Given a convex polytope of a set of points , any element belonging to it and not belonging to the convex hull of all its points but itself is a vertex.

$$s \in S \ : \ s \in Ext(\mathcal{P}_S) \iff s \notin \mathcal{P}_{S\setminus\{s\}} \tag{2.1.22}$$

Thanks to the next theorem we justify the above definition and characterize a convex polytope in terms of its vertices inequivocably.

Theorem 2.11 (Points form a Polytope)**.** *A convex polytope is the convex hull of its vertices.*

$$\mathcal{P}_S \equiv \boldsymbol{C}_{(Ext(\mathcal{P}_S))} = \boldsymbol{C}_{S'} \ : \ S' = Ext(\mathcal{P}_S) \tag{2.1.23}$$

Proof. Consider S, where $|S| \geq n$ and such that $Ext(\mathcal{P}_S) = \{s_1, \dots, s_n\} = S' \subseteq S$ are its vertices. Proving that $\mathbf{C}_S \subseteq \mathbf{C}_{S'}$ is sufficient for the claim. The proof is carried out by induction. **Base Case:** For $|S| = n$ this is trivially verified as the sets coincide.
Induction Hypothesis: assume the size of S is now strictly greater than n. Namely, $|S| > n = |S'|$
Conclusion: having that $|S| > |S'|$ then at least one element s^* is not in S'. By not being a vertex, it can be expressed as a linear combination of the remaining terms:

$$s^* = \sum_{s_i \in S, s_i \neq s^*} \xi_i s_i \ : \ \sum_i \xi_i = 1 \tag{2.1.24}$$

By theorem 2.7 we then know that any point in the convex polytope $x \in \mathcal{P}_S$ can be expressed as a linear combination of its members as:

$$x = \sum_{s_i \in S, s_i \neq s^*} \lambda_i s_i + \lambda^* s^* \tag{2.1.25}$$

$$= \sum_{s_i \in S, s_i \neq s^*} \lambda_i s_i + \lambda^* \sum_{s_i \in S, s_i \neq s^*} \xi_i s_i \tag{2.1.26}$$

$$= \sum_{s_i \in S, s_i \neq s^*} (\lambda_i + \lambda^* \xi_i) s_i \tag{2.1.27}$$

Where the sum of the weights is unitary as it is a convex combination. Given that the sum is of n - 1 elements it means that the size - 1 convex hull is included in the n size convex hull, this can be done for any size $> n$, exploiting the fact that: $\sum_{s_i \in S, s_i \neq s^*} (\lambda_i + \lambda^* \xi_i) = 1$ □

Definition 2.12 (Totally Unimodular Matrices $\mathcal{A}$)**.** Totally unimodular matrices have each square submatrix with determinant either $0, 1, -1$.

$$\mathcal{A} \coloneqq \left\{ A \in \mathfrak{M}_{n,E} \mid \forall q < n, q < E, \ det(A[r_1, \dots, r_q, c_1, \dots c_q]) \in \{0, 1, -1\} \right\} \tag{2.1.28}$$

Where by $A[\dots, \dots]$ we denote a square submatrix with some columns and some rows.

Total unimodularity is the matrix counterpart of bipartite graphs, thus characterizing disjoint sets when represented as linear maps.

Theorem 2.13 (Total Unimodularity of Bipartite Graphs)**.** *The incidence matrix of a bipartite graph is Totally Unimodular.*

$$B \in \mathscr{B} \ : \ A \ incidence \iff A \in \mathcal{A} \tag{2.1.29}$$

Proof. **($\implies$ direction)** Since edges are in the columns, and each edge joins two vertices, each column has either two ones or all zeros.
By induction on a general graph $B \in \mathscr{B}$, consider all its square submatrices A' of size k.
Base case: for $k = 1$ the determinant is trivially either 0 or 1.
Induction Hypothesis: Assume it is true $\forall A' \ : \ A' \in \mathfrak{M}_{k-1,k-1}$
Conclusion: let $A' \in \mathfrak{M}_{k,k}$ be a submatrix of A. Since it is a submatrix, then it must be the case that for each column j' there are either all zeros, a single non zero entry, or two non zero entries. The middle case happens if we select in the rows only one of the two vertices joined by an edge in the columns. The cases are analyzed below.
If A' has a column with all zeros (i.e. we are considering an edge that is not incident to any of the vertices in the rows), then the determinant is zero.

$$if \ \exists j' \ : \ \sum_{j'} a_{i'j'} = 0 \implies det(A') = 0 \tag{2.1.30}$$

If A' has a column with one non zero entry at coordinates $i'j'$, then $det(A') = \pm det(A'')$ where A'' is the submatrix obtained removing row i' and column j'. By $A'' \in \mathfrak{M}_{k-1,k-1}$, the induction hypothesis holds.

$$if \ \exists j' \ : \ \sum_{j'} a_{i'j'} = 1 \implies det(A') = \pm \det(A'') \ : \ A'' \in \mathfrak{M}_{k-1,k-1} \implies det(A') \in \{0, 1, -1\} \tag{2.1.31}$$

Lastly, if A' has all entries with two ones in the columns, we have a sub bipartite graph originated from B. Thanks to its properties, we consider a partition of the rows of A' into the two disjoint vertex sets. This division guarantees that the two have column sum equal to one, as any edge joins vertices of the two sets. Remembering that the determinant of a matrix is zero if and only if the rows are linearly dependent, by summing up all the rows in one vertex set and subtracting those of the other we get zero, thus $det(A'') = 0$.

$$A' \ incidence \ B' = \{(\mathscr{V}' \cup \mathscr{W}'), \mathscr{E}'\} \tag{2.1.32}$$

$$\implies A' = \left[A'_{\mathscr{V}'} \mid A'_{\mathscr{W}'} \right] \tag{2.1.33}$$

$$\implies \sum_{i' \in \mathscr{V}'} a_{i'j'} - \sum_{i' \in \mathscr{W}'} a_{i'j'} = 0 \implies det(A') = 0 \tag{2.1.34}$$

($\impliedby$ direction) Let $A \in \mathcal{A}$, suppose A generates a graph $G \in \mathfrak{G}$. By contradiction assume that G is not bipartite, i.e. $G \notin \mathscr{B}$.
G must contain a cycle of odd length[1] k. Taking the submatrix A' of this path indexed by $\{v_1, \dots v_k\} \times \{e_1, \dots, e_k\}$, assuming that column transformations lead to the ordered

[1] Clearly this is the case as there is at least one edge violating the inside disjoint vertex sets assumption

path from the first to the penultimate vertex, it will be that:

$$A' = \begin{bmatrix} 1 & 1 & 0 & \dots & \dots & 0 & 0 \\ 0 & 1 & 1 & \dots & \dots & 0 & 0 \\ 0 & 0 & 1 & \dots & \dots & 0 & 0 \\ \dots & \dots & \dots & \ddots & & \dots & \dots \\ \dots & \dots & \dots & & \ddots & \dots & \dots \\ 0 & 0 & 0 & \dots & \dots & 1 & 1 \\ 1 & 0 & 0 & \dots & \dots & 0 & 1 \end{bmatrix} \tag{2.1.35}$$

Intuitively, first and last connect, but also first and second, and so on.
It can be shown that $det(A') = 2 \implies A \notin \mathcal{A}$, which contradicts the assumption. □

The fact that bipartite incidence matrices are Totally Unimodular is of crucial importance. Indeed, in Definition 1.17 the incidence matrix came up in a formulation of $\mathfrak{P}$. The following Corollary lays the ground for what will be used in the next chapters.

Theorem 2.14 (Solutions are integers)**.** *The solution to a feasible linear system $Ax = b$ where A is Totally Unimodular is such that x is integral.*

$$A \in \mathcal{A} \ : det(A) \neq 0, \ Ax = b \implies x \in \{0,1\}^d \tag{2.1.36}$$

Proof. Feasibility is ensured by $det(A) \neq 0$ and using Cramer's rule:

$$Ax = b \iff x = A^{-1}b \iff x_i = \frac{det(A_i)}{det(A)} \ \forall i \in \{1, \dots, d\} \tag{2.1.37}$$

Where A_i is used to denote the matrix with the i^{th} column replaced by b.
By A being Totally Unimodular all numerators are either integer or null, and all denominators are integers. The solution is integral.

$$\implies x_i \in \{0,1\}^d \forall i \tag{2.1.38}$$

□

In the event that the reader wanted to solve the exact problem, Theorem 2.14 guarantees that a valid candidate would be found. However, in practice, this is not the case as the exact solution is intractable, while the relaxed solution is tractable. For the purpose of this task, a famous theorem by Hoffman and Kruskal [4] provides a formal solution, of which the easier proof by Veinott and Dantzig [5] is proposed. This result is carried out in two steps, the former is a theorem, which has as a corollary Hoffman and Kruskal's result.

Theorem 2.15 (Equivalent Properties of Integer Matrices)**.** *If a matrix A is integral*[2] *and has linearly independent rows, then the following are equivalent:*

1. *The determinant of every basis is either $+1$ or -1*
2. *The extreme points of the system $\{x : Ax = b, \ x \geq 0\}$ are integral for all integral b*
3. *Every basis has an integral inverse*

[2] meaning that its entries are only integers

Proof. $(1 \implies 2)$ Apply Theorem 2.14
$(2 \implies 3)$ Let B be a basis, and $\mathbf{y} \in \mathbb{N}^n$ be such that:

$$\mathbf{z} = \mathbf{y} + B^{-1}\mathbf{1}_i \geq 0 \implies B\mathbf{z} = B\mathbf{y} + \mathbf{1}_i = b$$

Which is integral by assuming point 2. For this reason, being B a basis of the space, z is itself integral. We then have that:

$$\mathbf{z} - \mathbf{y} = B^{-1}\mathbf{1}_i \in \mathbb{N}^n \ \ \forall i$$

Being that every index i is integral, the matrix B^{-1} is integral.
$(3 \implies 1)$ Let B be an integral basis and B^{-1} its integral inverse. Then both their determinants are integral and non zero.

$$\implies det(B) \in \mathbb{N} \ \ det(B^{-1}) \in \mathbb{N} \tag{2.1.39}$$

By basic linear algebra it holds that:

$$det(B)det(B^{-1}) = 1 \implies det(B) = det(B^{-1}) = \pm 1$$

Having proved a cycle of implications claims 1, 2 & 3 are equivalent. □

Corollary 2.16 (Hoffman and Kruskal's Theorem)**.** *If A is an integral matrix, the following are equivalent:*

1. *A is totally unimodular, $A \in \mathcal{A}$*
2. *If b is integral the extreme points of $\{x : Ax \leq b,\ x \geq 0\}$ are integral*

$$b \in \mathbb{N}^n \implies s \in Ext\left\{\mathcal{Z}_{A,b} \bigcap \{x \geq 0\}\right\} : s \in \mathbb{N}^n \tag{2.1.40}$$

Proof. Let $A' = \left[I_n \mid A\right]$ be so that the identity matrix is stacked with A. Then A' is integral and the rows are linearly independent thanks to the addition of I. It is possible to apply Theorem 2.15 as underlying properties and recreate a cycle.
$(1,\ thm2.15 \iff 1)$ Let $A \in \mathcal{A}$ be such that every basis has determinant $\in \{\pm 1\}$. Take a submatrix of A denoted as A_{sub} of rank $n - k$ and build by permuting the rows the following basis:

$$B = \begin{bmatrix} A_{sub} & 0_{n-k,n-k} \\ D & I_k \end{bmatrix}$$

Then $A \in \mathcal{A} \implies det(A_{sub}) \in \{0, \pm 1\} \iff det(B) \in \{0, \pm 1\}$. Thus condition 1 of Theorem 2.15 is equivalent to the first condition.
$(2 \implies 1)$ Let $\mathcal{Z}_{A,b} \cap \{x \geq 0\}$ have integral extreme points. Consider A', $b \in \mathbb{R}^n$ and the polyhedron:

$$\mathcal{Q} = \{z | z \geq 0 \ A'z = b\} \tag{2.1.41}$$

It has integer extreme points by Theorem 2.15. Noticing that $z = \begin{bmatrix} x \\ x' \end{bmatrix}$ where $x \in \mathbb{R}^E$ and $x' \in \mathbb{R}^n$. Thus

$$b = Ax + x' \implies x' = b - Ax \tag{2.1.42}$$

(Subproof of extremeness) Then, x is an extreme point of $\mathcal{Z}_{A,b} \bigcap \{x \geq 0\} = \{x | x \geq 0 \ Ax \leq b\}$. Otherwise, by contradiction assume that $x = \frac{1}{2}(v+w)$ where $v, w \in \mathcal{Z}_{A,b}$. Then it holds that:

$$x' = b - Ax = b - \frac{1}{2}A(v+w) = \frac{1}{2}(b - Av) + \frac{1}{2}(b - Aw) \tag{2.1.43}$$

$$\implies z = \begin{bmatrix} x \\ x' \end{bmatrix} = \frac{1}{2}\begin{bmatrix} b - Av \\ v \end{bmatrix} + \frac{1}{2}\begin{bmatrix} b - Aw \\ w \end{bmatrix} \tag{2.1.44}$$

$$\implies z \notin Ext(\mathcal{Q}) \tag{2.1.45}$$

Reaching a contradiction as z would not be a vertex of $\mathcal{Q}$.

Thus, x is a vertex of $\{x | x \geq 0, Ax \leq b\}$ which by assumption is integer. Then $x' = b - Ax$ is integer as well. By Theorem 2.15 the matrix $A' = \begin{bmatrix} I_n \mid A \end{bmatrix}$ is totally unimodular, which implies that A is totally unimodular as well. □

2.2 Duality: some equivalent results

It is also useful to introduce a different view on the problem, which arises from its double face. For this purpose, we introduce the objects that follow.

Definition 2.17 (Cover $\mathscr{C}$)**.** Given $G = (\mathscr{V}, \mathscr{E}) \in \mathfrak{G}$, a cover $\mathscr{C}$ is a set of vertices such that each edge has one endpoint included in it.

$$\mathscr{C} \subseteq \mathscr{V} \ : \ \forall e = (v, w) \ v \in \mathscr{C} \vee w \in \mathscr{C} \tag{2.2.1}$$

In literature and standard notation, its minimum size is denoted as $\tau(G)$. A cover of minimum size is said to be **perfect**.

Definition 2.18 (Fractional Cover)**.** Given $G = \{\mathscr{V}, \mathscr{E}\} \in \mathfrak{G}$ and a weight function $C : \mathscr{E} \to \mathbb{Z}_+$ a fractional vertex cover is a function $\kappa(\cdot)$ where:

$$\kappa : \mathscr{V} \to \mathbb{Z}_+ \tag{2.2.2}$$

$$\forall e \in \mathscr{E} \ \kappa(v) + \kappa(w) \leq C(v, w) \tag{2.2.3}$$

$$\mathrm{K}(G) = \sum_{v \in \mathscr{V}} \kappa(v) \text{ is the weight} \tag{2.2.4}$$

Definition 2.19 (Minimum size maximum fraction Vertex Cover problem $\mathfrak{D}$)**.** Given a graph $G = (\mathscr{V}, \mathscr{E}) \in \mathfrak{G}$, with cost function C representing vertex weights, find its minimum size maximum fraction perfect cover. With uniform costs, this is equivalent to finding the minimum size perfect cover.

It can be shown that finding the minimum vertex cover is $NP - complete$ for a general graph $G \in \mathfrak{G}$. For a bipartite graph instead, many equivalent and important results guarantee its existance and link $\mathfrak{D}$ with $\mathfrak{P}$. The nature of this link arises from duality theory in linear programming.
Before doing so, we introduce some results from [6], which proposes a nice framework to deal with this topic. First of all, the easy direction of the inequality that is to be proved is proposed. Then, the most difficult direction is derived from the famous work of Kőnig and Egerváry.
This allows to enter the realm of a set of equivalent results which characterize a Bipartite Graph and its geometry.

name	symbol	items	requirement	best size
matching	$\mathscr{M}$	edges	at most one edge per vertex	$\nu(G)$
edge cover	$\mathcal{M}$	edges	at least one edge per vertex	$\rho(G)$
vertex cover	$\mathscr{C}$	vertices	at least one vertex per edge	$\tau(G)$
stable set	$\mathcal{C}$	vertices	at most one vertex per edge	$\alpha(G)$

Table 2.1: Edge-Vertex definitions for a graph G

Definition 2.20 (Stable Set $\mathcal{C}$)**.** Given a graph $G = \{\mathscr{V}, \mathscr{E}\} \in \mathfrak{G}$, a stable set is a subset of non adjacent vertices:

$$\mathcal{C} \subseteq \mathscr{V} \ : \ e \nsubseteq \mathcal{C} \forall e \in \mathscr{E} \tag{2.2.5}$$

Its maximum size is denoted as $\alpha(G)$

Lemma 2.21 (Stable sets and vertex covers)**.** *Let $G = \{\mathscr{V}, \mathscr{E}\} \in \mathfrak{G}$ then:*

$$\mathcal{C} \textit{ stable} \iff \mathscr{V} \setminus \mathcal{C} \textit{ vertex cover} \tag{2.2.6}$$

Proof. **($\implies$ direction)** From $\mathcal{C}$ stable there are no adjacent vertices in $\mathcal{C}$. Thus, not all edges are necessarily "touched", and anyway if they are "touched" this happens once. Taking all those elements that are not in $\mathcal{C}$ a cover $\mathscr{C}$ will be obtained as all edges will be "touched" by a vertex in $\mathscr{V} \setminus \mathcal{C}$.
($\impliedby$ direction) If $\mathscr{C}$ is a cover, consider $\mathscr{V} \setminus \mathscr{C}$. This set is a set of vertices that necessarily does not include "joined" vertices, as in $\mathscr{C}$ at least one vertex was selected for each edge, so in $\mathscr{V} \setminus \mathscr{C}$ we have that:

- If two vertices $v, w \in \mathscr{C} : e = (v, w) \implies$ *none selected in* $\mathscr{V} \setminus \mathscr{C}$
- If one vertex $v \in \mathscr{C} : w \notin \mathscr{C} \forall e \in \mathscr{E} \ : \ e = (v, w) \implies w \in \mathscr{V} \setminus \mathscr{C}, w \notin \mathscr{V} \setminus \mathscr{C}$[3]

□

Definition 2.22 (Edge Cover $\mathcal{M}$)**.** Given a graph $G = \{\mathscr{V}, \mathscr{E}\} \in \mathfrak{G}$ an edge cover is a collection of edges such that for each vertex there is at least one edge containing it.

$$\mathcal{M} \subseteq \mathscr{E} \ : \ \forall v \in \mathscr{V} \exists e \in \mathcal{M} : v \in e \tag{2.2.7}$$

Its minimum size is denoted as $\rho(G)$

For the sake of completeness, Table 2.1 summarizes the features of the objects in Definitions 1.13, 2.17, 2.20, 2.22.

Lemma 2.23 (Easy Cover and Matching inequalities, weak duality)**.** *Given a graph $G = \{\mathscr{V}, \mathscr{E}\} \in \mathfrak{G}$ it holds that:*

$$\alpha(G) \leq \rho(G) \tag{2.2.8}$$
$$\nu(G) \leq \tau(G) \tag{2.2.9}$$

Proof. Fix a graph G. For simplicity let $\mathscr{E} \neq \emptyset$.
(max stable set vs min edge cover) Consider the maximum size stable set, assume its size is $\alpha(G)$. This is the maximum number of not adjacent vertices. Clearly then $\alpha(G) < |\mathscr{V}|$ as at least one vertex has to be ignored. Whatever the graph, the max stable set size indicates the maximum number of vertices not joined by an edge.
Consider a stable set $\mathcal{C}$ of maximum size $|\mathcal{C}| = \alpha(G) = k$. Then, $\alpha(G) \leq n$ as it cannot

[3]This is not complete as a mathematical explanation as we could have loopy joins of three vertices in a triangle, but it works nevertheless by reasoning. The vertices NOT in a cover form a stable set.

contain all vertices. Any edge cover $\mathcal{M}$, having at least one edge per vertex, has at least $|\mathcal{M}| = \rho(G) \geq n$ elements. Thus $\rho(B) \geq \alpha(G)$
(max matching vs min vertex cover) Consider the optimal size matching $\mathscr{M}$ such that $\nu(G) = |\mathscr{M}| = k$. Then, the $2k$ vertices in pairs are distinct, and any cover $\mathscr{C}$ must contain at least one node for each pair. Thus $|\mathscr{C}| \geq k = |\mathscr{M}|$. □

Theorem 2.24 (Gallai's Theorem)**.** *Let $G = \{\mathscr{V}, \mathscr{E}\} \in \mathfrak{G}$ then:*

$$\alpha(G) + \tau(G) = |\mathscr{V}| = \rho(G) + \nu(G) \tag{2.2.10}$$

Proof. The LHS and $|\mathscr{V}|$ are equal by Theorem 2.21.
For the LHS let $\mathscr{M} : |\mathscr{M}| = \nu(G)$. Then, the number of vertices that were not included is $|\mathscr{V}| - 2|\mathscr{M}|$. For each of these, add to $\mathscr{M}$ an incident edge e. The final size of the set is

$$size_{old} + size_{new} = |\mathscr{M}| + (|\mathscr{V}| - 2|\mathscr{M}|) = |\mathscr{V}| - |\mathscr{M}|$$

Now the set covers all edges, and it holds that:

$$\rho(G) \leq |\mathscr{V}| - \nu(G)$$

To prove the opposite direction, consider an edge covering $\mathcal{C} : |\mathcal{C}| = \rho(G)$. For each vertex $v \in \mathscr{V}$ delete delete **all but one** edges incident to v and inside $\mathcal{C}$, namely those that satisfy the double condition:

$$\delta_{\mathcal{C}}(v) \coloneqq \left\{(v, w) : w \in \delta(v) \wedge (v, w) \in \mathcal{C}\right\}$$

Until only one edge per vertex is left. Doing so, a matching $\mathscr{M}$ is obtained, with size:

$$\begin{aligned}
|\mathscr{M}| =& |\mathcal{C}| - \sum_{v \in \mathscr{V}} \delta_{\mathcal{C}} - 1 && \text{starting size minus deletion} \\
=& |\mathcal{C}| - \sum_{v \in \mathscr{V}} \delta_{\mathcal{C}}(v) + \sum_{v \in \mathscr{V}} 1 && \text{where } \sum_{v \in \mathscr{V}} 1 = |\mathscr{V}| \\
=& |\mathcal{C}| - \sum_{v \in \mathscr{V}} \delta_{\mathcal{C}}(v) + |\mathscr{V}| && \text{where } \sum_{v \in \mathscr{V}} \delta_{\mathcal{C}}(v) \leq 2|\mathcal{C}| \\
\geq& |\mathcal{C}| - 2|\mathcal{C}| + |\mathscr{V}| \\
=& |\mathscr{V}| - |\mathcal{C}|
\end{aligned}$$

Which eventually proves the opposite direction of the inequality, resulting in:

$$\begin{cases} \rho(G) \leq |\mathscr{V}| - \nu(G) \\ \nu(G) \geq |\mathscr{V}| - \rho(G) \end{cases} \implies \rho(G) + \nu(G) = |\mathscr{V}| = \alpha(G) + \tau(G)$$

□

Lemma 2.25 (Relaxed minimization program is integer)**.** *Let $A \in \mathcal{A}, \boldsymbol{b} \in \mathbb{R}^n, \boldsymbol{c} \in \mathbb{R}^E$. Then:*

$$min\{y^T \boldsymbol{b} | y \geq 0, y^T A \geq \boldsymbol{c}^T\} \in \mathbb{Z}^n \tag{2.2.11}$$

Proof. It is sufficient to notice that:

$$A \in \mathcal{A} \iff \begin{bmatrix} -I \\ A^T \\ -A^T \end{bmatrix} \in \mathcal{A} \tag{2.2.12}$$

So that it is possible to rearrange the integer (**NOT** relaxed) minimization problem as

$$\begin{bmatrix} -I \\ A^T \\ -A^T \end{bmatrix} y \leq \begin{bmatrix} 0 \\ +\mathbf{c} \\ -\mathbf{c} \end{bmatrix} \tag{2.2.13}$$

Using Theorem 2.15 and Corollary 2.16 the optimums of the relaxed formulation are integers. □

Theorem 2.26 (Kőnig Theorem)**.** *Given a Bipartite graph B the size of its maximum matching $\nu(B)$ is equal to the size of its minimum cover $\tau(B)$.*

$$B = \{(\mathscr{V} \cup \mathscr{W}), \mathscr{E}\} \in \mathscr{B} \implies \nu(B) = \tau(B) \tag{2.2.14}$$

Proof. For a short proof it is worth mentioning [7].
From Lemma 2.23 Equation 2.2.9 one direction was already proved. It suffices to show that:

$$B \in \mathscr{B} \implies \nu(B) \geq \tau(B) \tag{2.2.15}$$

Nevertheless, reasoning in equality terms is slightly easier. In matrix form the equality translates to:

$$max\{\mathbf{1}^T x | x = 0 \; Ax \leq \mathbf{1}\} = min\{\mathbf{y}^T \mathbf{1} | y \geq 0 \; y^T A \geq \mathbf{1}\} \tag{2.2.16}$$

Where the RHS is an expression of the vertex cover problem in its linearly relaxed form. By Corollary 2.16 and Lemma 2.25 the solutions to these optimization problems are integers. The constraints also induce them in the $\{0,1\}$ space element-wise[4].

$$x^* \in \{0,1\}^E \;\; y^* \in \{0,1\}^n \tag{2.2.17}$$

Considering $\mathscr{M} = \{e | x_e^* = 1\} \subseteq \mathscr{E}$ it holds that:

1. It is a matching (at most one edge per vertex)

$$\forall v \in \mathscr{V} \nexists e_1, e_2 \in \mathscr{E} \; : \; v \in e_1 \wedge v \in e_2 \wedge e_1, e_2 \in \mathscr{M} \tag{2.2.18}$$

2. It has cardinality equal to the objective function evaluated at the solution, which is maximal

$$|\mathscr{M}| = \mathbf{1}^T x^* = \nu(G) \tag{2.2.19}$$

Considering $\mathscr{C} = \{v | y_v^* = 1\} \subseteq \mathscr{V}$ it holds that:

1. It is a vertex cover (at least one vertex per edge)

$$\forall e \in \mathscr{E} \exists v \in \mathscr{C} \; : \; v \in e \tag{2.2.20}$$

2. It has cardinality equal to the objective function evaluated at the solution, which is minimal

$$|\mathscr{C}| = y^{*T} \mathbf{1} = \tau(B) \tag{2.2.21}$$

In the above numbered lists points 2 guarantee that the values considered are both optimal , while points 1 imply that the matching has at most one edge per vertex, and the cover has at least one vertex per edge. This logically implies that:

$$\nu(B) = \tau(B) \tag{2.2.22}$$

□

[4]For the vertex cover, suppose a component is integer but not in $\{0,1\}$. Then, by decreasing it to 1, the two constraints are anyway satisfied and the solution is lower in value, thus it can never happen that there are values other than $\{0,1\}$ for the latter problem.

Kőnig's Theorem is an antecedent of Duality, and is indeed a specific case of this great result in Polyhedral Combinatorics. Having had lectures on the topic, the reader is rerouted to another source [6] [8], . What follows is a collection of results specifically needed for the aims of the document, which presume basic knowledge of duality.

This Theorem relates non weighted versions of the two problems, but is crucial to observe the easy case. Indeed, further advancements thanks to Jenő Egerváry lead to the more general statement for weighted instances reported below as a corollary. It is notably presented in its max match min cover version, but this can be flipped easily to find the minim cost matching.

Corollary 2.27 (Kőnig-Egerváry theorem, strong duality)**.** *Let $B = \{(\mathscr{V}, \mathscr{W}), \mathscr{E}\} \in \mathscr{B}$ and consider an edge cost function $C : \mathscr{V} \times \mathscr{V} = \mathscr{E} \to \mathbb{Z}+$. Then: the maximum weight of a matching is equivalent to the minimum value of*

$$max\{\sum_{e \in \mathscr{M} x_e c_e} \} = min\{\sum_v f(v)\} : f \in \{g : \mathscr{V} \to \mathbb{Z}_+ \;\; g(u) + g(v) \geq c_{uv} \; \forall (u,v) \in \mathscr{E}\} \tag{2.2.23}$$

Proof. Using the linear programming formulation it is possible to obtain Statement 2.2.24

$$max\{\mathbf{c}^T x | x \geq 0 \quad Ax \leq \mathbf{1}\} = min\{y^T \mathbf{1} | y \geq 0 \; y^T A \geq \mathbf{c}\} \tag{2.2.24}$$

By the graph being bipartite, Theorem 2.13 ensures that the incidence matrix is totally unimodular. Then, Corollary 2.16 and Lemma 2.25 ensure that the solutions of such linear programs are integers. The former is a matching, the latter is a cover, on the same graph B, represented through the matrix $A \in \mathcal{A}$ by Theorem 2.13. Thus, by Kőnig's Theorem it is possible to assert that $\nu(B) = \tau(B)$ and this concludes the proof. Namely:

$$\begin{cases} \overset{Cor2.16}{\underset{Lem2.25}{\Longrightarrow}} x \in \mathbb{Z}^E, y \in \mathbb{Z}^n \\ A \in \mathcal{A} \overset{Thm2.13}{\Longleftrightarrow} B \in \mathscr{B} \\ \overset{Thm2.26}{\Longrightarrow} \nu(B) = \tau(B) \end{cases} \implies max\{\mathbf{c}^T x | Ax \leq \mathbf{1}\} = min\{y^T \mathbf{1} | y \geq 0, y^T A \geq \mathbf{c}^T\} \tag{2.2.25}$$

□

Observation 2.28 (On $\mathfrak{P}$ and $\mathfrak{D}$)**.** *Using primal-dual transformations, it is clear that an instance of the assignment problem $\mathfrak{P}$ is a primal linear program with respect to its dual vertex cover problem in $D \in \mathfrak{D}$.*

Assumption 2.29 (Duality Notation)**.** The purpose of using $\mathfrak{P}$ and $\mathfrak{D}$ was to exactly match the usual notation.
To align with the usual approach, the primal is a maximization problem, in our case, it would be nice to minimize the cost instead. For this reason, when showing fundamental properties, the notation will be kept as in other sources, while when dealing with the specific minimum cost bipartite matching, the extremals will be flipped. It is known that up to a polynomially bounded number of operations, the forms are equivalent.
In linear programming form an instance P is expressed through the following optimization problem:

$$max\{\mathbf{c}^T x\} \tag{2.2.26}$$

subject to: (2.2.27)

$$(P) \qquad Ax \leq \mathbf{b} \;\; where \;\; \mathbf{b} = \mathbf{1} \qquad x \geq 0 \tag{2.2.28}$$

In linear programming form an instance D is expressed through the following optimization problem:

$$min\{\mathbf{b}^T y\} \qquad where\ \ \mathbf{b} = \mathbf{1} \tag{2.2.29}$$

$$\text{subject to:} \tag{2.2.30}$$

$$(D) \qquad A^T y \geq \mathbf{c} \tag{2.2.31}$$

$$y \geq 0 \tag{2.2.32}$$

For the purpose of proposing a solution to the assignment problems $\mathfrak{P}$, an important concept is complementary slackness.

Theorem 2.30 (Complementary Slackness)**.** *Let $\mathscr{M}$ be feasible for $P \in \mathfrak{P}$, and $\mathscr{C}$ be feasible for its dual $D \in \mathfrak{D}$. The graph is the same $B \in \mathscr{B}$. Then:*

$$|\mathscr{M}| = \nu(B) \wedge |\mathscr{C}| = \tau(B) \iff \begin{cases} y_v = 0 \vee \sum_{e \in \mathscr{E}} A_{ve} x_e = b_v & \forall v \in \{1, \dots, n\} \\ x_e = 0 \vee \sum_{v \in \mathscr{V}} A^T_{ev} y_v = \boldsymbol{c}_e & \forall e \in \{1, \dots, E\} \end{cases} \tag{2.2.33}$$

Which is equivalent to asserting:

$$y_v(b_v - \sum_{e \in \mathscr{E}} A_{ve} x_e) = 0 \quad \forall v \in \mathscr{V} \tag{2.2.34}$$

$$x_e(\boldsymbol{c}_e - \sum_{v \in \mathscr{V}} A^T_{ev} y_v) = 0 \quad \forall e \in \mathscr{E} \tag{2.2.35}$$

Proof. If the primal and the dual program have feasible solutions at $\mathscr{M}$ and $\mathscr{C}$ this is equivalent to saying that the following conditions hold altoghether:

$$\begin{cases} Ax \leq \mathbf{b} & x \geq 0 \\ A^T y \geq \mathbf{c} & y \geq 0 \end{cases} \tag{2.2.36}$$

By crossing the conditions the requirements can be expressed as:

$$\implies \begin{cases} x \cdot \mathbf{c} = x^T \mathbf{c} \leq x^T A^T y \\ y \cdot \mathbf{b} = y^T \mathbf{b} \geq y^T A x \end{cases} \tag{2.2.37}$$

Where $x \in \mathbb{R}^E$, $A \in \mathfrak{M}_{n,E}$, $y \in \mathbb{R}^n$, which implies that:

$$\implies x^T A^T y = y^T A x \implies x \cdot \mathbf{c} \leq x^T A^T y = y^T A x \leq y \cdot \mathbf{b} \tag{2.2.38}$$

To prove sufficiency and necessity, it is possible to prove a chain of $\iff$ statements, which is as follows:

$$max\{\mathfrak{P}\} = min\{\mathfrak{D}\} \qquad \text{strong duality holds} \tag{2.2.39}$$

$$\iff x \cdot \mathbf{c} = y \cdot \mathbf{b} \qquad \text{by equality in optimal sizes} \tag{2.2.40}$$

$$\iff x^T \mathbf{c} = x^T A^T y = y^T A x = y^T \mathbf{b} \qquad \text{by Equation 2.2.38} \tag{2.2.41}$$

$$\iff \begin{cases} x \cdot (A^T y - \mathbf{c}) = 0 \\ y \cdot (Ax - \mathbf{b}) = 0 \end{cases} \qquad \text{rearranging} \tag{2.2.42}$$

Elaborating on the first equality result (the second follows with the same arguments), it can be added that:

$$\iff \begin{cases} x \cdot (A^T y - \mathbf{c}) = 0 \iff & \sum_e \left(x_e (\sum_v A^T_{ev} y_v - c_e) \right) = 0 \\ x_e \geq 0 \forall e & \\ A^T y - \mathbf{c} \geq 0 \iff & \sum_v A^T_{ev} y_v - c_e \geq 0 \end{cases} \tag{2.2.43}$$

And by element wise positivity (an implication of the feasibility assumption, third condition in the system), this implies that the equality is zero element wise $\forall e$, as the es in the sum cannot cancel each other out by being non negative. Hence:

$$\iff y_v = 0 \vee \sum_{e \in \mathscr{E}} A_{ve} x_e = b_v \quad \forall v \in \{1, \dots, n\} \tag{2.2.44}$$

Again, the proof for the second result follows by the same arguments on the other two feasibility conditions. □

These results, embedded in the framework of the matching problem, lead to graph specific interpretations of this duality, as shown in the next arguments.

Definition 2.31 (Saturating Matching in a bipartite graph)**.** Given $B = \{(\mathscr{V}, \mathscr{W}), \mathscr{E}\} \in \mathscr{B}$ a $\mathscr{V}$-saturating matching is a matching $\mathscr{M}$ such that all the all the vertices of $\mathscr{V}$ are covered.

Theorem 2.32 (Hall's Marriage Theorem)**.** *Given $B = \{(\mathscr{V}, \mathscr{W}), \mathscr{E}\} \in \mathscr{B}$ there is a $\mathscr{V}$ saturating matching if and only if:*

$$|\delta(V)| \geq |V| \forall V \subseteq \mathscr{V} \tag{2.2.45}$$

Where by $\delta(\cdot)$ the neighbors of all the elements of the set V are considered.

Proof. There are independent proofs, but one exploiting previous results is proposed.
Assume B has no matching saturating $\mathscr{V}$. By Theorem 2.26 There is a cover $\mathscr{C}$ such that $|\mathscr{C}| < |\mathscr{V}|$.
Consider $V = \mathscr{V} \bigcap \mathscr{C}$ and $W = \mathscr{W} \bigcap \mathscr{C}$, the disjoint sets of vertices making the cover. It trivially holds that:

$$V \bigcup W = \mathscr{C} \implies |V| + |W| = |\mathscr{C}| < |\mathscr{V}| \tag{2.2.46}$$

$$\implies |W| < |\mathscr{V}| - |V| = |\mathscr{V} \setminus V| \tag{2.2.47}$$

Moreover, by $\mathscr{C}$ being a cover, it is also the case that there are no edges between $\mathscr{V} \setminus V$ and $\mathscr{W} \setminus W$. This is given by the fact that a cover must include at least a vertex per each edge. Thus, while in general for bipartite graphs it holds that:

$$|\delta(\mathscr{V} \setminus V) \leq |\mathscr{W}| \tag{2.2.48}$$

The condition that there is not saturating matching is updated, resulting in:

$$|\delta(\mathscr{V} \setminus V)| \leq |W| < |\mathscr{V} \setminus V| \tag{2.2.49}$$

By choosing as set $\mathscr{V} \setminus V$ Hall's condition is not satisfied. □

Thanks to this theorem, any instance of the problem $\mathfrak{P}$ can be solved, once transformed in its $\mathscr{K}_{n,n}$ form.

Corollary 2.33 (Existance of solutions of $\mathfrak{P}$)**.** *For any instance of the bipartite matching problem there is a solution.*

$$\forall P \in \mathfrak{P} \exists \mathscr{M}^* \text{ optimal} \tag{2.2.50}$$

Proof. Consider the formulation of P in graph form as in Definition 1.14, where B denotes its bipartite graph representation. Build its complete bipartite graph $\mathscr{K}_{n,n}$. By construction it holds that:

$$|\mathscr{V}| = |\mathscr{W}| = n \tag{2.2.51}$$
$$|\delta(v)| = n \forall v \in \mathscr{V} \tag{2.2.52}$$
$$|\delta(w)| = n \forall w \in \mathscr{W} \tag{2.2.53}$$

Clearly, any subset of m vertices has more neighbors than elements for each of the two vertex sets. It holds bidirectionally that there exists a $\mathscr{V}$-saturating matching and a $\mathscr{W}$-saturating matching, which coincide as all the elements are considered and the size is n for both. □

Observation 2.34 (Φ can be ignored)**.** *This last result allows to ignore the Φ feasibility function of the Combinatorial Optimization Problem formulation of Definition 1.3. Indeed, any permutation does not need to be checked and is trivially a valid matching, as long as the permutation is a shuffle of indices of $\mathscr{A}$ agents and $\mathscr{T}$ tasks.*

Going back to Double stochasticity, Polytopes and the equivalence between linear and integer programming, these last results allow us to strentghten even more the results of Kőnig's theorem.

Lemma 2.35 (Doubly stochastic matrices preserved by convex combination)**.** *A convex combination of doubly stochastic matrices is doubly stochastic.*

$$\{X^{(d)}\}_{d=1}^{D} : X^{(d)} \in \mathscr{X} \subset \mathfrak{M}_{n,n} \quad \{\lambda^{(d)}\}_{d=1}^{D} : \lambda^{(d)} \in \mathbb{R} \forall d, \sum_{d} \lambda^{(d)} = 1 \implies \sum_{d=1}^{D} \lambda^{(d)} X^{(d)} \in \mathscr{X} \tag{2.2.54}$$

Where the $^{(d)}$ notation is used to avoid confusing between indices of the entries and elements of the combination.

Proof. By Definition 2.1 the row and column sum for each matrix is unitary, thus:

$$\forall d, i, j \quad \sum_{i=1}^{n} X_{ij}^{(d)} = 1 \quad \sum_{j=1}^{n} X_{ij}^{(d)} = 1 \tag{2.2.55}$$

Moreover, all the elements are positive. Let $X^* = \sum_{d=1}^{D} \lambda^{(d)} X^{(d)}$ and inspect its entries to check if these conditions hold as well. As a first observation, all the entries are positive by being a sum of positive entries. Secondly, for the rows:

$$X_{k\cdot}^* = \sum_{l=1}^{n} X_{kl}^* \tag{2.2.56}$$
$$= \sum_{l=1}^{n} \left(\sum_{d=1}^{D} \lambda^{(d)} X^{(d)} \right)_{kl} \tag{2.2.57}$$
$$= \sum_{d=1}^{D} \lambda^{(d)} \left(\sum_{l=1}^{n} X_{kl}^{(d)} \right) \qquad \text{as } \lambda^{(d)} \perp\!\!\!\perp l, k \tag{2.2.58}$$
$$= \sum_{d=1}^{D} \lambda^{(d)} \cdot 1 \qquad \text{by Equation 2.2.55} \tag{2.2.59}$$
$$= 1 \cdot 1 \forall k \; rows \qquad \text{by convex combination} \tag{2.2.60}$$

The same can be done for the columns, rearranging indices. Thus:

$$X^* = \sum_{d=1}^{D} \lambda^{(d)} X^{(d)} \in \mathscr{X} \tag{2.2.61}$$

□

Theorem 2.36 (Birkhoff Von Veumann Theorem)**.** *Every doubly stochastic matrix is expressible as a convex combination of permutation matrices and viceversa a convex combination of permutation matrices is doubly stochastic.*
Equivalently, the polytope of permutation matrices is the convex hull of doubly stochastic matrices.
In matchings, every linearly relaxed solution is expressible as a combination of perfect matchings, which are then the vertices of the polytope where optimums are attained.

$$n \in \mathbb{N}, \mathcal{X} \subset \mathscr{X} \subset \mathfrak{M}_{n,n} \implies \boldsymbol{C}_{\mathcal{X}} = \mathscr{X} \tag{2.2.62}$$

Proof. **($\implies$ direction)** To prove that a convex combination of permutation matrices is doubly stochastic it suffices to use Lemma 2.35 and the fact that $\mathcal{X} \subset \mathscr{X}$. Thus, a convex combination of permutation matrices is a convex combination of doubly stochastic matrices which is a doubly stochastic matrix.
($\impliedby$ direction) The difficult direction is showing that the set of doubly stochastic matrices can be expressed as the convex hull of permutations. Let $X \in \mathscr{X}$. X can be interpreted as a bipartite graph $B = \{(\mathscr{V} \cup \mathscr{W}), \mathscr{E}\}$ with $\mathscr{V}$ representing the rows, $\mathscr{W}$ representing the columns and $\mathscr{E}$ formed by the nonzero entries of X, with cost(weight) $X_{ij} > 0$.
Consider a $v \in \mathscr{V}$ and its neighbors $\delta(v) \subseteq \mathscr{W}$. Similarly $\delta(w) \subseteq \mathscr{V} \forall w \in \mathscr{W}$. Moreover by $X \in \mathscr{X}$:

$$\forall w \sum_{w \in \delta(v)} X_{vw} = \sum_{j=1}^{n} X_{ij} = 1 \tag{2.2.63}$$

$$\forall v \sum_{v \in \delta(w)} X_{vw} = \sum_{i=1}^{n} X_{ij} = 1 \tag{2.2.64}$$

$$\implies \sum_{w \in W} \sum_{v \in \delta(w)} X_{vw} = \sum_{w \in W} 1 = |W| \tag{2.2.65}$$

$$\implies \sum_{v \in V} \sum_{w \in \delta(v)} X_{vw} = \sum_{v \in V} 1 = |V| \tag{2.2.66}$$

Considering instead $\delta(V)$ it holds that $V \subseteq \delta(\delta(V))$ and it is possible to assert:

$$|\delta(V)| = \sum_{w \in \delta(V)} \sum_{v \in \delta(w)} X_{vw} \geq \sum_{w \in \delta(V)} \sum_{v \in V} X_{vw} = |V| \tag{2.2.67}$$

$$\implies |\delta(V)| \geq |V| \tag{2.2.68}$$

$$\overset{Thm 2.32}{\implies} \exists \mathscr{M} \subseteq \mathscr{E} : |\mathscr{M}| = n \; for \; B \tag{2.2.69}$$

Where in the last implication Hall's Theorem was used to state that a perfect matching can be found.
Considering the incidence matrix of the matching defined as:

$$\Pi : \begin{cases} \Pi_{ij} = 1 & if \; e = (i,j) \in \mathscr{M} \\ \Pi_{ij} = 0 & otherwise \end{cases} \tag{2.2.70}$$

(Subproof Π is a permutation matrix $\Pi \in \mathscr{X}$) consider $i \in \{1, \dots, n\}$, a row r_i of Π. Since $\mathscr{M}$ is perfect then:

$$\exists e \in \mathscr{M} : e = (r_i, c_j) \forall i, \ for\ some\ j \tag{2.2.71}$$
$$\implies \Pi_{ij} = 1 \tag{2.2.72}$$

By contradiction, suppose there are j, j' such that $\Pi_{ij} = \Pi_{ij'} = 1$, then:

$$\implies (r_i, c_j) \in \mathscr{M}\ (r_i, c_{j'}) \in \mathscr{M} \tag{2.2.73}$$
$$\implies r_i\ has\ two\ incident\ edges \tag{2.2.74}$$
$$\implies \mathscr{M}\ not\ valid \tag{2.2.75}$$

Thus for each row and column there is only one positive value, which implies $\Pi \in \mathscr{X}$.

Define $\lambda = \underset{i,j \in \{1,\dots,n\}}{min} \left\{ X_{ij} | \Pi_{ij} \neq 0 \right\}$. Clearly:

$$\lambda > 0 \tag{2.2.76}$$
$$\Pi_{ij} \neq 0 \implies X_{ij} \neq 0 \tag{2.2.77}$$
$$\lambda = X_{kl}\ for\ some\ (k, l) \tag{2.2.78}$$

Consider $Y = X - \lambda\Pi$, then:

$$Y = 0 \implies X = \lambda\Pi \implies \lambda = 1 \implies X = \Pi \implies X \in \mathscr{X} \implies proved \tag{2.2.79}$$
$$else \quad Y \neq 0 \tag{2.2.80}$$

In the first case, the original matrix X is a permutation matrix itself, in the latter case it is possible to add that:

$$\forall X_{ij} \neq 0 \quad \lambda\Pi_{ij} \leq \lambda \leq X_{ij} \implies Y_{ij} \geq 0 \forall (i, j) \tag{2.2.81}$$
$$Y_{kl} = X_{kl} - \lambda\Pi_{kl} = \lambda - \lambda \cdot 1 = 0 \tag{2.2.82}$$
$$X \in \mathscr{X}\ \Pi \in \mathscr{X} \tag{2.2.83}$$

Since every row and column of X and Π sums to one then any row or column of Y sums to $1 - \lambda$. Let $X' = \frac{1}{1-\lambda} Y$, then its rows and columns sum to one and $X' \in \mathscr{X}$. Rearranging the elements:

$$Y = X - \lambda\Pi \implies X = Y + \lambda\Pi = (1 - \lambda)X' + \lambda\Pi \ : \ X' \in \mathscr{X}, \Pi \in \mathscr{X} \tag{2.2.84}$$

So X is expressed as the convex combination of a doubly stochastic matrix and a permutation matrix. Moreover, the it holds that:

$$X_{ij} = 0 \implies \Pi_{ij} = 0 \implies X'_{ij} = 0 \implies |\{(i, j) : X'_{ij} = 0\}| \geq |\{(i, j) : X_{ij} = 0\}| \tag{2.2.85}$$
$$X_{kl} > 0 \tag{2.2.86}$$
$$X'_{kl} = \frac{1}{1-\lambda} Y_{kl} = \frac{1}{1-\lambda}(X_{kl} - \lambda\Pi_{kl}) = 0 \tag{2.2.87}$$

Namely, the zeros of the custom doubly stochastic matrix are strictly higher in number than the original matrix. The same can be done decomposing X'_{ij}, obtaining another permutation matrix and another doubly stochastic matrix, with more null entries. Considering that the maximum number of null entries is n^2, X will eventually be expressed

as a convex combination of at most n^2 permutation matrices, as the last element will necessarily be a permutation matrix, given that the minimum of only one entry is the entry itself.
It is eventually possible to confirm that:

$$\mathbf{C}_{\mathcal{X}} = \mathscr{X} \tag{2.2.88}$$

□

It can be proved ([9] for an incomplete but interesting reference) that Theorems 2.26, 2.27, 2.32 and 2.36 by Kőnig, Kőnig-Egerváry, Hall, and Birkhoff-Von Neumann, are all equivalent combinatorics statements, as well as some others not included in this document such as:

- Menger's Theorem
- Dilworth's Theorem
- The Max-flow Min-cut Theorem

This last result was actually part of the lectures in the Algorithms' graduate course offered at Bocconi University.

Chapter 3

The Hungarian Algorithm

We see the world in terms of our theories.

Thomas Kuhn

The original modern formulation of the Hungarian method was published by Harold Kuhn in 1955 [1]. It was not until 2016 that it was found that also Jacobi had proposed an equivalent method[10]. The term *Hungarian* is a reference to the research contributions of Kőnig and Egerváry . A nice overview of the history of such method is given in [10].

While Kuhn's formulation seems to exploit an intuitive observation with seemingly no formal meaning, all of his contribution can be encapsulated in a the framework of primal-dual algorithms. To introduce the procedure, the classic a black box formulation of the algorithm is proposed. Before doing so, the intuition and a couple assumptions are presented.

If done correctly, decreasing the cost of a row or a column does not contaminate the optimal arrangement.

Assumption 3.1 (On Observation 2.34)**.** As previously proved in Theorem 2.32, there is no need to include Φ in instances of $\mathfrak{P}$, as well as M which is always *min*. For this reason, an instance of the problem will be referred to as:

$$P = \Big(\mathscr{P} = \{\mathscr{A}, \mathscr{T}\}, C\Big) \in \mathfrak{P} \tag{3.0.1}$$

Moreover, also the specific form of C, thanks to Observation 1.20 is assumed to be that of a complete graph, already with modified edges and added vertices. Namely the cost function C is such that a complete bipartite graph $\mathscr{K}_{n,n}$ is induced, as well as a square matrix.

$$C : \mathscr{A} \times \mathscr{T} \to \mathbb{Z}_+ \ : \ |\mathscr{A}| = |\mathscr{T}| = n \tag{3.0.2}$$

Without loss of generality, agents a will be on the rows, and tasks t on the columns.

Remark (What is a line in a matrix?)**.** Given a matrix C by selecting or drawing a line the operation of highlighting a whole row or column is intended.

Thanks to the above adjustments, the operational formulation of Algorithm 1 is proposed. In Example 3.2 Algorithm 1 is applied to another instance, as the one proposed in Chapter 1 is *algorithmically* easy.

Algorithm 1 Black Box Hungarian Algorithm

Input: $P = (\mathscr{P}, C)$

1: row update: for each agent, find the lowest cost task and remove its value from the whole row
2: column update: for each task, findt the lowest cost agent and remove its value from the whole column
3: count and draw lines across columns or rows that include the fewest possible zeros.
4: **if** the number of lines is n **then**
optimal assignment $\leftarrow$ tuples (a_i, t_j) where one zero per row and per column
return optimal assignment
5: **else**
find lowest cost entry of the not selected c_{min}
6: **for** each not selected row $C_{i\cdot}$ **do**
$C_{i\cdot} \leftarrow C_{i\cdot} - c_{min}$
7: **end for**
8: **for** each selected n entry C_{ij} **do**
$C_{\cdot j} \leftarrow C_{\cdot j} + c_{min}$
9: **end for**
10: **end if**
11: Go back to step 4

Example 3.2 (Algorithm 1 in action)**.** Assume that for a problem $P \in \mathfrak{P}$ the cost matrix is of the form:

$$C = \begin{bmatrix} 108 & 125 & 150 \\ 150 & 135 & 175 \\ 122 & 148 & 250 \end{bmatrix} \tag{3.0.3}$$

So iterations go as follows:

Step 1

$$C \rightsquigarrow \begin{bmatrix} 0 & 17 & 42 \\ 150 & 135 & 175 \\ 122 & 148 & 250 \end{bmatrix} \rightsquigarrow \begin{bmatrix} 0 & 17 & 42 \\ 15 & 0 & 40 \\ 122 & 148 & 250 \end{bmatrix} \rightsquigarrow \begin{bmatrix} 0 & 17 & 42 \\ 15 & 0 & 40 \\ 0 & 26 & 128 \end{bmatrix} \tag{3.0.4}$$

Step 2

$$\begin{bmatrix} 0 & 17 & 42 \\ 15 & 0 & 40 \\ 0 & 26 & 128 \end{bmatrix} \rightsquigarrow \begin{bmatrix} 0 & 17 & 42 \\ 15 & 0 & 40 \\ 0 & 26 & 128 \end{bmatrix} \rightsquigarrow \begin{bmatrix} 0 & 17 & 2 \\ 15 & 0 & 0 \\ 0 & 26 & 88 \end{bmatrix} \tag{3.0.5}$$

Step 3

$$\begin{bmatrix} 0 & 17 & 2 \\ 15 & 0 & 0 \\ 0 & 26 & 88 \end{bmatrix} \rightsquigarrow \begin{bmatrix} 0 & 17 & 2 \\ 15 & 0 & 0 \\ 0 & 26 & 88 \end{bmatrix} \implies \text{two lines} \tag{3.0.6}$$

Step 4 There are $2 < n$ lines

Step 5 $c_{min} = 2$

Step 6

$$\begin{bmatrix} 0 & 17 & 2 \\ 15 & 0 & 0 \\ 0 & 26 & 88 \end{bmatrix} \rightsquigarrow \begin{bmatrix} -2 & 15 & 0 \\ 15 & 0 & 0 \\ -2 & 24 & 86 \end{bmatrix} \tag{3.0.7}$$

Step 8

$$\begin{bmatrix} -2 & 15 & 0 \\ 15 & 0 & 0 \\ -2 & 24 & 86 \end{bmatrix} \rightsquigarrow \begin{bmatrix} 0 & 15 & 0 \\ 17 & 0 & 0 \\ 0 & 24 & 86 \end{bmatrix} \tag{3.0.8}$$

Back to Step 3

$$\begin{bmatrix} 0 & 15 & 0 \\ 17 & 0 & 0 \\ 0 & 24 & 86 \end{bmatrix} \rightsquigarrow \begin{bmatrix} 0 & 15 & 0 \\ 17 & 0 & 0 \\ 0 & 24 & 86 \end{bmatrix} \implies \text{three lines} \tag{3.0.9}$$

Step 4 There are $3 = n$ lines.

$$\begin{bmatrix} 0 & 15 & 0 \\ 17 & 0 & 0 \\ 0 & 24 & 86 \end{bmatrix} \rightsquigarrow \begin{bmatrix} 0 & 15 & 0 \\ 17 & 0 & 0 \\ 0 & 24 & 86 \end{bmatrix} \rightsquigarrow \star = \{(a_3, t_1), (a_2, t_2), (a_1, t_3)\} \tag{3.0.10}$$

At the advantage of not requiring any theoretical knowledge, the impression is that Algorithm 1 is an inexplicable black box. Yet, tricks always have a reason, and the next sections will justify convergence, efficiency and where properties are applied with a flow perspective, following the approach of [11].

3.1 Unravelling the procedure

Definition 3.3 (Symmetric difference operation Δ)**.** Given two sets A, B the symmetric difference of them returns elements that are in A but not in B or in B but not in A. It is a logical equivalent to the XOR $\oplus$ operation.

$$A \Delta B \coloneqq (A \setminus B) \cup (B \setminus A) = \{x \ : \ (x \in A) \oplus (x \in B)\} \tag{3.1.1}$$

Lemma 3.4 (Properties of Δ)**.** *The symmetric difference operation has the following useful properties:*

1. $A \Delta B = B \Delta A$
2. $A \Delta B = (A \cup B) \setminus (A \cap B)$
3. $A \Delta B = (A \cap B^C) \cup (B \cap A^C)$

Definition 3.5 (Alternating Path or Route $\mathcal{R}_{\mathscr{M}}$)**.** Given a graph $G \in \mathfrak{G}$ and a matching $\mathscr{M} \subseteq \mathscr{E}$ an alternating path of vertices is a path that indeed alternates between edges $e \in \mathscr{M}$ and edges $e \in \mathscr{V} \setminus \mathscr{M}$.

Definition 3.6 (Augmenting Path or Route)**.** An augmenting path is an alternating path where the first and the last vertices are exposed, so no edges in $\mathscr{M}$ are incident to the endpoints of the path.

Lemma 3.7 (General properties of an alternating path in a bipartite graph)**.** *Given a bipartite graph $B \in \mathscr{B}$, a matching $\mathscr{M}$ and an augmenting path $\mathcal{R}_{\mathscr{M}}$ it holds that:*

1. *if $\mathcal{R}_{\mathscr{M}}$ has k edges in $\mathscr{M}$ then it has $k+1$ edges in $\mathscr{V} \setminus \mathscr{M}$*
2. *the endpoints of $\mathcal{R}_{\mathscr{M}}$ are on different vertex sets of the bipartite graph*
3. *The symmetric difference of a matching and its augmenting path is a matching with $+1$ size*

$$\mathscr{M} \Delta \mathcal{R}_{\mathscr{M}} = \mathscr{M}' \text{ where } |\mathscr{M}'| = |\mathscr{M}| + 1 \tag{3.1.2}$$

Proof. **(Claim 1)** By induction on k, given a bipartite graph B and a matching $\mathscr{M}$.
Base case: for $k = 0$ there are no edges in $\mathscr{M}$, an augmenting path must have at least two vertices (otherwise it is not a path), and contain one vertex from $\mathscr{V}$ and one from $\mathscr{W}$ as the edges are present only across the disjoint sets. If there are no edges from the matching, then a valid augmenting path has only one edge from the unmatched ones,

joining two vertices which are not incident to any of the edges of $\mathscr{M}$.
Inductive hypothesis: assume it is true for a general k number of edges from $\mathscr{M}$.
Conclusion: Given $k+1$ edges from $\mathscr{M}$ the aim is to show that for a valid augmenting path there are $k+2$ edges not from $\mathscr{M}$. Any augmenting path must start and end with vertices which are not incident to any edge in $\mathscr{M}$. Fixing the endpoints, for each $\mathcal{R}_{\mathscr{M}}$ the last and the first edge will necessarily not come from $\mathscr{M}$. This proves the claim, as for k + 1 edges, augmenting, there will be exactly k edges not from $\mathscr{M}$ augmenting those $k+1$ inside, plus 2 to make the endpoints valid for an augmenting path outside.
(Claim 2) if one endpoint v is in $\mathscr{V}$ and by Claim one there are $2k+1$ edges in the augmenting path, then the number of jumps is odd. Given that the Bipartite graph B has only edges connecting vertices of $\mathscr{V}$ to vertices of $\mathscr{W}$ the second endpoints must be in $\mathscr{W}$.
(Claim 3) Taking the definition of symmetric difference:

$$\mathscr{M}\Delta\mathcal{R}_{\mathscr{M}} = (\mathscr{M} \setminus \mathcal{R}_{\mathscr{M}}) \cup (\mathcal{R}_{\mathscr{M}} \setminus \mathscr{M}) \tag{3.1.3}$$

It is worth noticing that the two resulting sets are disjoint. The former contains edges of the matching which were not selected. The latter contains the edges which are not in $\mathscr{M}$ from the augmenting path.
Without loss of generality, let $|\mathscr{M}| = L$, where $k < L$ edges[1] where included in $\mathcal{R}_{\mathscr{M}}$.
First from Claim 1 it holds that:

$$|\mathscr{M}'| = |\mathscr{M} \setminus \mathcal{R}_{\mathscr{M}}| + |\mathcal{R}_{\mathscr{M}} \setminus \mathscr{M}| = (L-k) + (k+1) = L + 1 > |\mathscr{M}| \tag{3.1.4}$$

So the size increases. Secondly, consider the union of these two sets is composed of those edges that were forming a matching and were not included in the path, and those edges from the path not included in the matching. Clearly they are disjoint. Moreover, if some edge e was matching a vertex v, and was included in the path, this is not part of the new set, i.e. $e \notin \mathscr{M}\Delta\mathcal{R}_{\mathscr{M}}$, thus the edge is not matched thanks to e. Nevertheless, it is matched by the adjacent edge $e' \in \mathcal{R}_{\mathscr{M}}$ which satisfies the augmenting requirement and is clearly not part of $\mathscr{M}$, namely:

$$\forall v \textit{ matched by some } e \in \mathscr{M} \implies \exists e' \in \mathcal{R}_{\mathscr{M}},\ e' \notin \mathscr{M} \textit{ incident to } v \implies e' \in \mathscr{M}' \tag{3.1.5}$$

Thus, $\mathscr{M}'$ is a matching. □

The importance of such a result is that provided with an augmenting path, it is possible to improve the size of a given matching. When such an operation is done, it will be denoted as augmenting $\mathscr{M}$ along $\mathcal{R}_{\mathscr{M}}$. It also turns out that augmenting paths inequivocably characterize the validity of a perfect matching, as the Theorem by Berge [12] below shows.

Theorem 3.8 (Berge's Theorem)**.** *For a bipartite graph $B \in \mathscr{B}$ a matching $\mathscr{M}$ is of maximum cardinality if and only if there are no augmenting paths with respect to it.*

$$\mathscr{M} \ :\ |\mathscr{M}| = \nu(B) \iff \nexists \mathcal{R}_{\mathscr{M}} \textit{ augmenting} \tag{3.1.6}$$

Proof. **($\implies$ direction)** By contradiction assume $\mathscr{M}$ is of maximum cardinality but there is an augmenting path. By Lemma 3.7, there is a matching $\mathscr{M}' = \mathscr{M}\Delta\mathcal{R}_{\mathscr{M}}$ which is of higher cardinality. This contradicts the fact that $\mathscr{M}$ itself is of maximum cardinality.
($\impliedby$ direction) By contradiction on $\not\Longrightarrow$, assume $\mathscr{M}$ is not of maximum cardinality,

[1] The corner case where $L = k$ will be treated separately in Theorem 3.8 below.

and that there are no alternating paths. Consider $\mathscr{M}^*$ to be of maximum cardinality. Let $\mathscr{L} = \mathscr{M} \Delta \mathscr{M}^*$. Then,

$$|\mathscr{L}| = |\mathscr{M} \setminus \mathscr{M}^*| + |\mathscr{M}^* \setminus \mathscr{M}| \tag{3.1.7}$$

Where $|\mathscr{M}^* \setminus M| > |\mathscr{M} \setminus \mathscr{M}^*|$ as $|\mathscr{M}^*| > |\mathscr{M}|$ by the optimal being greater in size. Moreover, the symmetric difference creates alternations of edges of $\mathscr{M}$ and $\mathscr{M}^*$ as in the proof of Claim 3 of Lemma 3.7. Thus, $\mathscr{L}$ is made of paths and cycles that alternate edges of the two sets. Cycles will have even length, not breaking the advantage in size of $\mathscr{M}^*$. Instead, cycles have the peculiarity that there are more elements from the maximum cardinality matching. Therefore, there is at least one augmenting path, contradicting the inexistance assumption. □

The process of finding subsequent $\mathcal{R}_{\mathscr{M}}$ can be formalized in by introducing Directed Graphs and building a custom directed Graph, as follows.

Definition 3.9 (Directed Graphs $\mathfrak{Q}$)**.** Directed graphs are objects composed of edges and vertices where edges join vertices and have a direction. According to some rule, the direction will be expressed as positivity or negativity of the edge.

$$\mathfrak{Q} \coloneqq \left\{ Q = \{\mathscr{V}, \mathscr{E}\} \ : \ e \in \mathscr{E} \textit{ paired with } \pm \right\} \tag{3.1.8}$$

If a weight function is provided, the convention follows the same approach.

Definition 3.10 (Directed Bipartite Graphs $\mathscr{Q}$)**.** A directed bipartite graph is a bipartite graph with disjoint sets and directed edges. Weighted versions follow the same fashion.

$$\mathscr{Q} \subseteq \mathfrak{Q} \quad \mathscr{Q} \coloneqq \left\{ Q = \{(\mathscr{V}, \mathscr{W}), \mathscr{E}\} \ : \ e \in \mathscr{E} \textit{ paired with } \pm \right\} \tag{3.1.9}$$

For the purpose of finding an augmenting path, a custom directed graph will be exploited. Given a matching $\mathscr{M}$ it is build as follows:

- If the edge is in the matching $e \in \mathscr{M}$, then it goes from $\mathscr{W} \longrightarrow \mathscr{V}$
- If the edge is not in the matching $e \notin \mathscr{M}$, then it goes from $\mathscr{V} \longrightarrow \mathscr{W}$

By convention edges going from $\mathscr{V}$ to $\mathscr{W}$ are assumed to be positive, while on the opposite direction they are negative. As in Definition 3.10, weights are set according to positivity or negativity.

The advantage of using such a tool arises thanks to the Theorem below, which characterizes augmenting paths.

Theorem 3.11 (Augmenting path and directed graphs)**.** *Given a graph $G \in \mathfrak{G}$ and a matching $\mathscr{M}$ an augmenting path $\mathcal{R}_{\mathscr{M}}$ for the matching exists if and only if there is a directed path going from one exposed vertex in $\mathscr{W}$ to another one in $\mathscr{V}$.*

Proof. **($\implies$ direction)** Consider an augmenting path $\mathcal{R}_{\mathscr{M}}$, and the subsets of vertices:

$$V = \mathscr{V} \setminus \mathscr{V}_{\mathscr{M}} \quad W = \mathscr{W} \setminus \mathscr{W}_{\mathscr{M}} \tag{3.1.10}$$

Where the subscripts indicate the vertices exposed by edges in the matching.
By definition, $\mathcal{R}_{\mathscr{M}}$ has the endpoints in V and W. Similarly, a directed graph built as above can start from a vertex in W which is not exposed and end up in another not exposed vertex in V.
($\impliedby$ direction) If there is a directed graph going from an exposed vertex $w \in \mathscr{W}$ to

an exposed vertex $v \in \mathscr{V}$, then there are two edges not in the matching that join the endpoints of the path to exposed vertices. From these exposed vertices, the matching edges can be alternated to unmatched edges to form an alternating path. □

At the cost of finding augmenting paths, the unweighted assignment problem can be solved, by just iteratively using the subroutine presented in Algorithm 2

Algorithm 2 Max Cardinality Matching subroutine

Input: $P = (\mathscr{P})$ with equal negligible costs C, and a matching $\mathscr{M}$
Output: $|\mathscr{M}'| = |\mathscr{M}| + 1$

1: Build $Q \in \mathscr{Q}$ assigning:

$$\begin{cases} e^+ & if\ e \in \mathscr{M} \\ e^- & if\ e \notin \mathscr{M} \end{cases} \tag{3.1.11}$$

2: Find an augmenting path (a directed path) $\mathcal{R}_{\mathscr{M}}$ between:

$$V = \mathscr{V} \setminus \mathscr{V}_{\mathscr{M}} \quad W = \mathscr{W} \setminus \mathscr{W}_{\mathscr{M}} \tag{3.1.12}$$

3: $\mathscr{M}' \leftarrow \mathscr{M} \Delta \mathcal{R}_{\mathscr{M}}$
4: **return** $\mathscr{M}'$

With this procedure, it is possible to find efficiently a matching of maximum cardinality for an unweighted bipartite graph. The running time for a final solution can be bounded in terms of the size of the problem, and turns out being polynomial.

Theorem 3.12 (Bipartite Matching computational complexity)**.** *Given a bipartite graph $B \in \mathscr{B}$, running iteratively Algorithm 2 as a subroutine is bounded in terms of the size of the problem as:*

$$T(n) \in O(|\mathscr{V}||\mathscr{E}|) = O(n^3) \tag{3.1.13}$$

Proof. Finiteness of the procedure is ensured by the fact that:

- If the matching is not of maximum cardinality, there will always be exposed vertices from definition 1.13
- If there are exposed vertices, there is an augmenting path from Theorem 3.8
- The increase in size for each iteration is guaranteed to be 1 from Lemma 3.7

For these reasons, the subroutine will be called $O(|\mathscr{V}|) = O(n)$ times.
The running time of the subroutine is $O(|\mathscr{E}|)$ since to build a directed path it is required to scan all the edges checking if they belong to $\mathscr{M}$. Having formed the directed graph, finding an augmenting path takes time $O(\mathscr{E})$ as well. Moreover, by assumption 1.19 and 1.21 it holds that for any bipartite graph, whether complete or not $|\mathscr{E}| \leq n^2$. Thus:

$$T(n) \in O(|\mathscr{V}||\mathscr{E}|) = O(n^3) \tag{3.1.14}$$

□

In terms of the mirror dual problem, once no augmenting path is found, it is also possible to build the minimum vertex cover.

Definition 3.13 (Labeling set $\mathcal{L}_{\mathscr{M}}$)**.** Given a bipartite graph $B = \{(\mathscr{V}, \mathscr{W}), \mathscr{E}\} \in \mathscr{B}$ with matching $\mathscr{M}$, build its directed graph $Q \in \mathscr{Q}$. A labeling set is the set of vertices which can be reached with a directed path from an exposed vertex $v \in \mathscr{V} \setminus \mathscr{V}_{\mathscr{M}}$. By convention, it is the set of reachable vertices from one side of the graph through paths in the matching-induced directed graph.

Theorem 3.14 (Induced minimum size vertex cover)**.** *For a bipartite graph $B \in \mathscr{B}$, using the subroutine of Algorithm 2 until convergence returns a matching $\mathscr{M}$. Considering its linked labeling set $\mathcal{L}_{\mathscr{M}}$ it holds that:*

1. $\mathscr{C} = (\mathscr{V} \setminus \mathcal{L}_{\mathscr{M}}) \cup (\mathscr{W} \cap \mathcal{L}_{\mathscr{M}})$ *is a cover*
2. $|\mathscr{C}| = |\mathscr{M}|$ *Kőnig 's Theorem*

Proof. **(Claim** 1**)** By contradiction, let $\mathscr{C}$ not be a cover. Then, by definition, there is less than a vertex per edge. This in turn implies that there is an edge in the vertices not included in the cover. This set is the complement of the cover denoted as $\overline{\mathscr{C}} = (\mathscr{C})^C$. By using the definition of set difference $A \setminus B \coloneqq A \cap \overline{B}$ it is possible to derive:

$$\overline{\mathscr{C}} = \Big((\mathscr{V} \setminus \mathcal{L}_{\mathscr{M}}) \cup (\mathscr{W} \cap \mathcal{L}_{\mathscr{M}})\Big)^C \tag{3.1.15}$$

$$= \Big((\mathscr{V} \cap \overline{\mathcal{L}_{\mathscr{M}}}) \cup (\mathscr{W} \cap \mathcal{L}_{\mathscr{M}})\Big)^C \tag{3.1.16}$$

$$= (\mathscr{V} \cap \mathcal{L}_{\mathscr{M}}) \cup (\mathscr{W} \cap \overline{\mathcal{L}_{\mathscr{M}}}) \tag{3.1.17}$$

$$\implies \exists e = (v, w) \notin \mathscr{M} : v \in (\mathscr{V} \cap \mathcal{L}_{\mathscr{M}})\ w \in (\mathscr{W} \cap \overline{\mathcal{L}_{\mathscr{M}}}) \tag{3.1.18}$$

Where statement 3.1.18 follows from the edge existance (as $\mathscr{C}$ is not a cover and there is an edge in the complementary set), and the fact that the two sets are disjoint. Thus, in a bipartite graph, there must exist an edge connecting an element from the labelled subset of $\mathscr{V}$ and an element from the unlabelled subset of $\mathscr{W}$ which is **NOT** part of the matching.

(Subproof: *wts* $e \notin \mathscr{M}$**)** By contradiction, this very last fact is guaranteed since:

$$\textit{assume } e \in \mathscr{M} \implies w \in \mathcal{L}_{\mathscr{M}} \textit{ necessary for } v \in \mathcal{L}_{\mathscr{M}} \tag{3.1.19}$$

$$\implies \textit{contradiction with } w \in (\mathscr{W} \cap \overline{\mathcal{L}_{\mathscr{M}}}) \tag{3.1.20}$$

$$\implies e \notin \mathscr{M} \tag{3.1.21}$$

Namely, if one can be reached from the directed path, then also should the other to be part of the matching.

Thus, by construction (see Algorithm 2), Q is such that:

$$e \in (\mathscr{E} \setminus \mathscr{M}) \implies e_+ = (v, w)\ :\ v \to w \tag{3.1.22}$$

Following the notation introduced before. This also implies that $w \in \mathscr{W}$ can be reached from the exposed vertex $v \in \mathscr{V}$, through the edge $e = (v, w)$, which contradicts the fact that $w \in (\mathscr{W} \cap \mathcal{L}_{\mathscr{M}}) \implies w \notin \mathcal{L}_{\mathscr{M}}$. Therefore a contradiction is reached.
(Claim 2**)** It suffices to prove $|\mathscr{C}| \leq |\mathscr{M}|$ as the other direction was proved in Lemma 2.23. The same result was also proved in Theorem 2.26 but a different way of proving it is proposed in [11] and reported below.
Observing the following facts:

- No vertex in $\mathscr{W} \cap \mathcal{L}_{\mathscr{M}}$ is exposed, as otherwise there would be an augmenting path and no maximum cardinality matching by Theorem 3.8
- By definition of $\mathcal{L}_{\mathscr{M}}$, no vertex in $\mathscr{V} \setminus \mathcal{L}_{\mathscr{M}}$ is exposed
- No edge is found between them as proved in Claim 1 above.

Therefore, every vertex in $\mathscr{C}$ is matched, with distinct edges, thus $|\mathscr{C}| \leq |\mathscr{M}|$ resulting again in the more general result that:

$$|\mathscr{C}| \leq |\mathscr{M}| \implies \tau(B) = |\mathscr{C}| = |\mathscr{M}| = \nu(B) \tag{3.1.23}$$

□

While this result is of pivotal importance, it is not yet clear how it generalizes to weighted instances of $\mathfrak{P}$.

Recalling Definition 2.18, and Theorem 2.36 a fractional cover can be interpreted as the counterpart of edge costs in a dual intepretation. From a graph perspective, it is often referred to as a **feasible labeling**. By construction, it is clear that for such object and a perfect matching $\mathscr{M}$ it holds that:

$$\sum_{e \in \mathscr{M}} \mathbf{c}_e = \sum_{i,j} c_{ij} \geq \sum_{i \in \mathscr{V}} \kappa(i) + \sum_{j \in \mathscr{W}} \kappa(j) \tag{3.1.24}$$

Which is the dual lower bound. If maximized, it would lead to the desired solution, attained exactly by strong duality. Indeed, considering a linear programming formulation for an instance $P \in \mathfrak{P}$ as in Definition 2.2:

$$\sum_{e=(v,w) \in \mathscr{E}} \mathbf{c}_e x_e = \sum_{i,j} \mathbf{c}_{ij} x_{ij} \geq \sum_{i \in \mathscr{V}} \sum_{j \in \mathscr{W}} (\kappa(i) + \kappa(j)) x_{ij} \tag{3.1.25}$$

$$\geq \sum_{i \in \mathscr{V}} \sum_{j \in \mathscr{W}} \kappa(i) x_{ij} + \sum_{i \in \mathscr{V}} \sum_{j \in \mathscr{W}} \kappa(j) x_{ij} \tag{3.1.26}$$

$$\geq \sum_{i \in \mathscr{V}} \left(\kappa(i) \sum_{j \in \mathscr{W}} x_{ij} \right) + \sum_{j \in \mathscr{W}} \left(\kappa(j) \sum_{i \in \mathscr{V}} x_{ij} \right) \tag{3.1.27}$$

$$\geq \sum_{i \in \mathscr{V}} \kappa(i) + \sum_{j \in \mathscr{W}} \kappa(j) \tag{3.1.28}$$

Where the last operation can be performed since the constraints of Definition 2.2 make the x_{ij} sum to unity, namely Equation 2.1.3.

These results can be summarized and extended with the following Theorem.

Theorem 3.15 (Duality, linear and integer programming)**.** *Given an instance of $P \in \mathfrak{P}$, and its bipartite graph $B \in \mathscr{B}$, the **decreasingly ordered** optimal solutions to the primal and dual formulation are:*

- *integer (minimum cost perfect matchings)*
- *linear (relaxed minimum cost)*
- *dual (fractional maximal cover).*

Namely:

$$\min_{\mathscr{M}:|\mathscr{M}|=\nu(B)} \left\{ \sum_{e=(i,j) \in \mathscr{M}} \boldsymbol{c}_e \right\} \geq \min_{x_{ij} \geq 0} \left\{ \boldsymbol{c}_{ij} x_{ij} \right\} \geq \max_{\kappa: \mathscr{V} \text{ or } \mathscr{W} \to \mathbb{Z}_+} \left\{ \sum_{i \mathscr{V}} \kappa(i) + \sum_{j \in \mathscr{W}} \kappa(j) \right\} \tag{3.1.29}$$

Where the first has integral constraints, $x_e \in \{0,1\} \forall e$, the second uses relaxed constraints that sum to unity, and the third is the dual representation where κ might map from either of the two distinct vertex sets to positive integer numbers.

Proof. Though already proved across the document with Theorem 2.36, Equation 3.1.27 and Observation 2.3. Here, they will be shown in order to orient the reader. Considering

a solution $x^* \in \mathbb{R}^E$ to the integer problem of Definition 1.10, a relaxed solution $x^*_{rel} \in \mathbb{R}^E$ from Definition 2.2 and a dual $y^* \in \mathbb{R}^n$ solution of Definition 2.19 we have that the first represents a valid matching of minimum cost, the second is a solution to the same problem but relaxed and the third is a solution to the dual problem. Then

$$\overset{Obs\ 2.3}{\Longrightarrow} \quad \min_{\mathscr{M}:|\mathscr{M}|=\nu(B)} \left\{ \sum_{e=(i,j)\in\mathscr{M}} \mathbf{c}_e \right\} \geq \min_{x_{ij}\geq 0} \left\{ \mathbf{c}_{ij} x_{ij} \right\} \tag{3.1.30}$$

$$\overset{\substack{Thm\ 2.36 \\ Eqn\ 3.1.27}}{\Longrightarrow} \quad \min_{x_{ij}\geq 0} \left\{ \mathbf{c}_{ij} x_{ij} \right\} \geq \max_{\kappa:\mathscr{V} \text{ or } \mathscr{W}\to\mathbb{Z}_+} \left\{ \sum_{i\mathscr{V}} \kappa(i) + \sum_{j\in\mathscr{W}} \kappa(j) \right\} \tag{3.1.31}$$

$$\Longrightarrow \quad \min_{\mathscr{M}:|\mathscr{M}|=\nu(B)} \left\{ \sum_{e=(i,j)\in\mathscr{M}} \mathbf{c}_e \right\} \geq \min_{x_{ij}\geq 0} \left\{ \mathbf{c}_{ij} x_{ij} \right\} \geq \max_{\kappa:\mathscr{V} \text{ or } \mathscr{W}\to\mathbb{Z}_+} \left\{ \sum_{i\mathscr{V}} \kappa(i) + \sum_{j\in\mathscr{W}} \kappa(j) \right\} \tag{3.1.32}$$

□

In the specific setting of looking for an Algorithm solving weighted instances of $\mathfrak{P}$, this result is crucial. Indeed, provided that a feasible solution to $D \in \mathfrak{D}$ is found, linked to a perfect matching $\mathscr{M}$ would lead to the $\geq$ signs to become $=$, a case of complementary slackness from Theorem 2.30, which using the framework introduced is equivalent to asserting:

$$\begin{cases} \sum_{e\in\mathscr{M}} \mathbf{c}_e = \sum_{i\in\mathscr{V}} \kappa(i) + \sum_{j\in\mathscr{W}} \kappa(j) \\ \mathbf{c}_e = \mathbf{c}_{ij} \geq 0 \\ \kappa(i) \geq 0 \\ \kappa(j) \geq 0 \end{cases} \tag{3.1.33}$$

Which by the same element wise positivity arguments implies that:

$$\Longrightarrow \ \omega_{ij} = \mathbf{c}_{ij} - \kappa(i) - \kappa(j) = 0 \tag{3.1.34}$$

Whenever a dual solution such that $\omega_{ij} = 0 \forall (i,j) \in \mathscr{M}$ is found, the algorithm can return with an optimal matching. If this is perfect, iterations to improve the proposed feasible dual are implemented until the perfect size is reached.

Definition 3.16 (Slack weight function $\omega(\cdot)$)**.** Consider a bipartite graph $B \in \mathscr{B}$ with primal $P \in \mathfrak{P}$ and dual $D \in \mathfrak{D}$ both feasible. The slack weight function is a function checking the complementary slackness condition for a given edge.

$$\omega : \mathscr{E} \to \mathbb{Z}_+ \ \ \omega_{ij} = \mathbf{c}_{ij} - \kappa(i) - \kappa(j) \ \forall e = (i,j) \in \mathscr{E} \tag{3.1.35}$$

Where equivalent notations are $\omega_e = \omega_{ij} = \omega(e) = \omega(i,j)$

Thanks to the above results, Algorithm 1 can be viewed in terms of easier to understand graph operations. For any perfect matching found, there will be a *dual* vertex cover recovered through a labeling as in Definition 3.13 and guaranteed to be optimal using Theorem 3.14. Finding an optimal procedure to update feasible dual solutions and reach a perfect matching is the only missing piece. The next section outlines the order of the steps, and presents a version in the broader space of primal-dual algorithms.

3.2 Graph Version, Primal-Dual Version

Given the operational nature of the methods, most of the results will be proposed in plain text.

The starting point is a dual feasible solution, which counts as a lower bound. It is rather easy to assert that for a given bipartite graph $B \in \mathscr{B}$ a *trivial* feasible solution is:

$$D_{start} = \begin{cases} \kappa(i) = 0 & \forall i \in \mathscr{V} \\ \kappa(j) = \underset{i \in \mathscr{V}}{min}\{\mathbf{c}_{ij}\} & \forall j \in \mathscr{W} \end{cases} \tag{3.2.1}$$

Using Theorem 2.30 the Dual, resulting in a cover, is optimal if and only if complementary slackness conditions hold, i.e. $\omega_{ij} = 0 \forall (i,j)$. This is verified where the exact condition that is wished for the whole graph is verified. Namely given a bipartite graph $B \in \mathscr{B}$ it is possible to build a subgraph B_ω as follows:

$$K_\omega \subseteq B \quad K_\omega \coloneqq \{(\mathscr{V} \cup \mathscr{W}), \mathscr{E}_\omega\} \quad where \quad \mathscr{E}_\omega = \{e = (i,j) \in \mathscr{E} \ : \ \omega_{ij} = 0\} \subseteq \mathscr{E} \tag{3.2.2}$$

Through Algorithm 2 a matching $\mathscr{M}$ for K_ω is recovered. If this matching is not of maximum cardinality (i.e. $|\mathscr{M}| = \nu(B) = n$), then the dual can be updated to get closer to the primal solution by changing the $\kappa(\cdot)$ function. Information on how and where to change is provided thanks to Theorem 3.14.

Considering $\mathcal{L}_\mathscr{M}$, it holds that:

$$\mathscr{C} = (\mathscr{V} \setminus \mathcal{L}_\mathscr{M}) \cup (\mathscr{W} \cap \mathcal{L}_\mathscr{M}) \implies \nexists e = (i,j) \in \mathscr{E}_\omega : i \in (\mathscr{V} \cap \mathcal{L}_\mathscr{M}),\ j \in (\mathscr{W} \setminus \mathcal{L}_\mathscr{M}) \tag{3.2.3}$$

Since edges not belonging to the cover are necessarily such that $\omega_{ij} > 0 \implies x_{ij} = 0$. For this reason, it is possible to devise a procedure to rebalance the choices of κ for elements of the two disjoint sets. The minimum weight edge through vertices not satisfying complementary slackness is:

$$\omega^* = \underset{i \in (\mathscr{V} \cap \mathcal{L}_\mathscr{M})\ j \in (\mathscr{W} \setminus \mathcal{L}_\mathscr{M})}{min} \{\omega_{ij}\} \tag{3.2.4}$$

And it is thus possible to move $\kappa(i)$ *away* from 0 further and further as:

$$\kappa(i) \leftarrow \kappa(i) + \omega^* \quad if \quad i \in (\mathscr{V} \cap \mathcal{L}_\mathscr{M}) \tag{3.2.5}$$

$$\kappa(j) \leftarrow \kappa(j) - \omega^* \quad if \quad j \in (\mathscr{W} \cap \mathcal{L}_\mathscr{M}) \tag{3.2.6}$$

Assuming that the dual for an iteration is feasible, it is rather straightforward to prove that such an update is feasible as well. It holds that $\kappa(\cdot)$ is positive, and the ω^* updates change the total cost as:

$$\sum_{new\ dual} \kappa - \sum_{old\ dual} \kappa = \omega^* \Big(|\mathscr{V} \cap \mathcal{L}_\mathscr{M}| - |\mathscr{W} \cap \mathcal{L}_\mathscr{M}| \Big) \qquad \text{weight} \times \text{added} - \text{removed} \tag{3.2.7}$$

$$= \omega^* \Big(|\mathscr{V} \cap \mathcal{L}_\mathscr{M}| + |\mathscr{V} \setminus \mathcal{L}_\mathscr{M}| - |\mathscr{V} \setminus \mathcal{L}_\mathscr{M}| - |\mathscr{W} \cap \mathcal{L}_\mathscr{M}| \Big) \qquad \text{adding and removing} \tag{3.2.8}$$

$$= \omega^* \Big(|\mathscr{V} \cap \mathcal{L}_\mathscr{M}| + |\mathscr{V} \cap \overline{\mathcal{L}_\mathscr{M}}| - |\mathscr{V} \setminus \mathcal{L}_\mathscr{M}| - |\mathscr{W} \cap \mathcal{L}_\mathscr{M}| \Big) \qquad \text{by def of } \setminus \tag{3.2.9}$$

$$= \omega^* (|\mathscr{V}| - |\mathscr{C}|) \qquad \text{by def of } \mathcal{L}_\mathscr{M} \text{ and } \mathscr{C} \tag{3.2.10}$$

$$= \omega^* (n - |\mathscr{C}|) \qquad \text{where } |\mathscr{C}| < n \tag{3.2.11}$$

Thus, the value of the dual strictly increases for each iteration. To complete the analysis, it suffices to show that the devised procedure terminates in finite time.

Theorem 3.17 (Graph Hungarian Finiteness)**.** *The graph hungarian version terminates in finite time.*

Proof. Consider a graph $B \in \mathscr{B}$ with current matching $\mathscr{M}$ for a given iteration. If a vertex $v \in \mathscr{V}$ is exposed, then no edge in $\mathscr{M}$ is incident to it. Thus, there is also a vertex $w \in \mathscr{W}$ which is reachable from v, otherwise, the edge joining them would be part of the matching. Now the cover can be constructed, and ω^* is necessarily found. This also implies that an edge between the cover induced sets (3.2.3, updated as in 3.2.5) is adjusted such that $\omega_{ij} = 0$ for some $e = (i, j)$ where $w_{ij} \equiv \omega^*$. This adds 1 vertex of $\mathscr{W}$ to those reachable from $v \in \mathscr{V}$ for each iteration. By the finiteness of vertices, the algorithm is finite, and the algorithm increases the size of $\mathscr{M}$ by at least one unit every n iterations.
Thus, the algorithm terminates in finite time. □

A preliminary analysis might indicate that the computational complexity of such a procedure is $T(n) \in O(n^4)$. In his publication, Kuhn only proved finiteness[1]. Few years later, Munkres extended previous results asserting that complexity was bounded by a cube factor of n, through reasonings on elementary matrix operations[13]. While following Munkres' approach could be instructive in terms of complexity analysis, it relies mostly on less visualizable considerations.

Instead, in terms of graph operations, it could be argued that:

- in n iterations either
  - all of $w \in \mathscr{W}$ are reachable (update dual condition)
  - $\mathscr{M}$ increases by 1
  - $\implies O(n^2)$ time in the worst case with 1 increase each n iterations.
- To compute $\mathcal{L}_{\mathscr{M}}$ it takes $O(n^2)$ time as it requires to scan all edges. This happens for each new $\mathscr{M}$ found.

Thus, it could be stated that:

$$T(n) \in O(n^4) \tag{3.2.12}$$

However, this is not as tight as it can get, as $\mathcal{L}_{\mathscr{M}}$ updates do not require to scan all edges again. An update in the matching size is found through an augmenting path incrementally, which means adding one vertex to the labeling, thus scanning only $O(n)$ vertices for each update. This ensures that again, as in 3.12 from Algorithm 2 for the unweighted version, computational time is:

$$T(n) \in O(n^3) \tag{3.2.13}$$

The adjusted procedure can be broken down into the following nested loops:

- In $O(n)$ outer iterations reach a perfect matching
- for the inner loop either of the two hold:
  - matching $O(n)$ to find, $O(n)$ to update
  - labeling $O(n)$ to find ω and update dual, at most $O(n)$ times.
- $\implies O(n^2)$ inner time in the worst case with 1 increase each n iterations.

Lastly, the whole procedure is outlined in Algorithm 3, to store all the steps in one place.

Algorithm 3 can also be viewed from a primal-dual perspective, using the approach of [14], outlined in Algorithm 4

Example 3.2 can be solved through this procedure.

Algorithm 3 Graph Hungarian Algorithm Version

Input: Instance $P = (\mathscr{P}, C) \in \mathfrak{P}$

Output: Minimum cost maximum cardinality bipartite matching $\mathscr{M}$

1: Build $K \in \mathscr{K}_{n,n}$ from P ▷ If necessary, adjust with Observation 1.20
2: Initialize D_{start} ▷ as in Equation 3.2.1
3: Evaluate $w_{ij} \forall e = (i,j) \in \mathscr{E}$ ▷ Complementary slackness, definition 3.16
4: Build the subgraph K_ω ▷ as in Equation 3.2.2
5: $\mathscr{M} \leftarrow$ max cardinality matching of K_ω ▷ Using subroutine Algorithm 2
6: **if** $|\mathscr{M}| == n$ **then** ▷ Perfect matching found
7: **return** $\mathscr{M}$
8: **end if**
9: Build $\mathcal{L}_{\mathscr{M}}$
10: Recover $\mathscr{C} \leftarrow (\mathscr{V} \setminus \mathcal{L}_{\mathscr{M}}) \cup (\mathscr{W} \cap \mathcal{L}_{\mathscr{M}})$ ▷ induced cover, Equation 3.2.3
11: Find $\omega^* \leftarrow \min_{i \in (\mathscr{V} \cap \mathcal{L}_{\mathscr{M}})} \min_{j \in (\mathscr{W} \setminus \mathcal{L}_{\mathscr{M}})} \{\omega_{ij}\}$ ▷ minimum weight, Equation 3.2.4
12: Update

$$\kappa(i) \leftarrow \kappa(i) + \omega^* \quad \forall \quad i \in (\mathscr{V} \cap \mathcal{L}_{\mathscr{M}})$$
$$\kappa(j) \leftarrow \kappa(j) - \omega^* \quad \forall \quad j \in (\mathscr{W} \cap \mathcal{L}_{\mathscr{M}})$$

▷ update rules, Equations 3.2.7, 3.2.8

13: Go back to **Step** 3

Algorithm 4 Primal - Dual Hungarian Algorithm Version

Input: Instance $P = (\mathscr{P}, C) \in \mathfrak{P}$

Output: Minimum cost maximum cardinality bipartite matching $\mathscr{M}$

```
Build K ∈ 𝒦_{n,n} from P
Dual initialization of κ                                   ▷ Step 2
Primal initialization of x                                 ▷ Steps 3, 4, 5
while x infeasible do                                      ▷ Step 6 not verified
    let 𝓡_𝓜 ← ∅                ▷ here 𝓜 is used for consistency, but this is x
    build ℒ_𝓜
    while 𝓡_𝓜 == ∅ do
        while 𝓡_𝓜 == ∅ ∧ ℒ_𝓜 ≠ ∅ do
            Update labels                                  ▷ Steps 9, 10
        end while
        if 𝓡_𝓜 == ∅ then
            Dual iteration                                 ▷ Steps 11, 12
        end if
    end while
    Primal iteration                                       ▷ Steps 3, 4, 5
end while
return x
```

Example 3.18 (Algorithm 3 in action)**.** Assume that the cost function is:

$$C = \begin{bmatrix} 108 & 125 & 150 \\ 150 & 135 & 175 \\ 122 & 148 & 250 \end{bmatrix} \tag{3.2.14}$$

It is possible to build the bipartite graph of Figure 3.1, where $\mathscr{A} \equiv \mathscr{V} = \{\alpha, \beta, \gamma\}$, and $\mathscr{T} \equiv \mathscr{W} = \{A, B, C\}$.

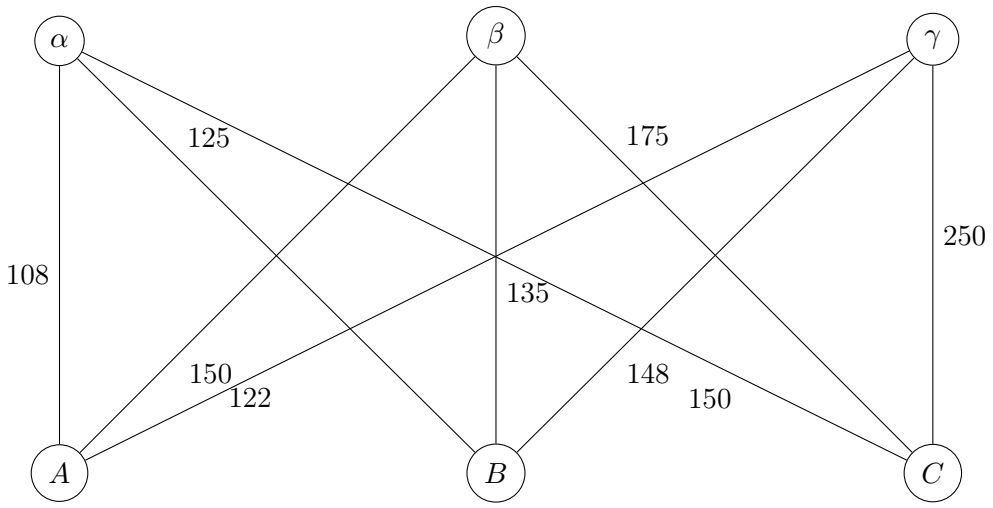


Figure 3.1: Bipartite graph

The dual is initialized as

$$\begin{cases} \kappa(i) = 0 & i = \alpha, \beta, \gamma \\ \kappa(A) = 108 \\ \kappa(B) = 125 \\ \kappa(C) = 150 \end{cases} \tag{3.2.15}$$

We evaluate ω_{ij} for all edges to check complementary slackness and find that it is null for $e \in \{(\alpha, A), (\alpha, B), (\alpha, C)\}$. Indeed, considering agents in the rows and tasks in the columns Tables 3.1, 3.2 are obtained.

	108	125	150
0	108	125	150
0	150	135	175
0	122	148	250

Table 3.1: Complementary slackness, $T = 1$

	108	125	150
0	0	0	0
0	42	10	25
0	14	23	100

Table 3.2: Complementary slackness weights $T = 1$

The subgraph K_ω is found on Figure 3.2.

A maximum cardinality matching $\mathscr{M} = \{(\alpha, C)\}$ is of size $|\mathscr{M}| = 1 < n = 3$. It is found using the subroutine of Algorithm 2 and is that of Figure 3.3:

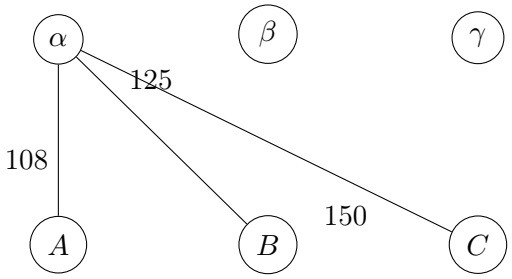


Figure 3.2: Subgraph $T = 1$

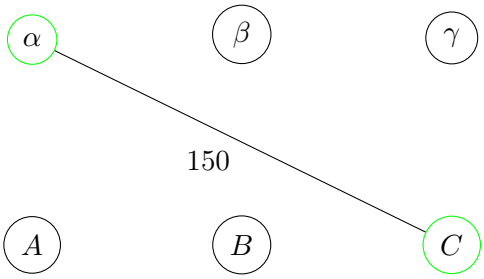


Figure 3.3: Max card $T = 1$

Then, its directed path is created in Figure 3.4

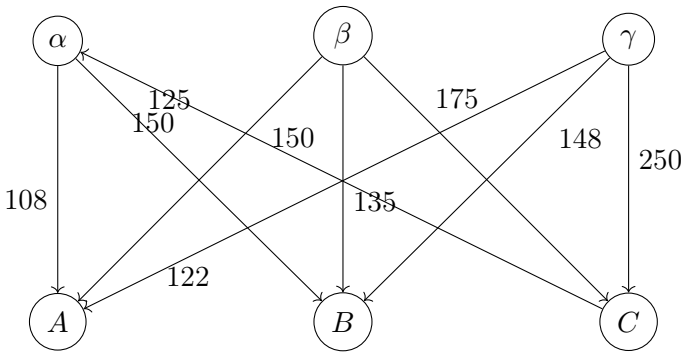


Figure 3.4: Directed graph $T = 1$

For this iteration, exposed vertices in $\mathscr{V}$ are $\{\beta, \gamma\}$. In $\mathscr{W}$ the exposed vertices are $\{A, B\}$. Using definition 3.13, the vertices reachable from exposed vertices in Q are

$$\mathcal{L}_{\mathscr{M}} = \{\beta, \gamma\} \tag{3.2.16}$$

A cover is built as previously explained as:

$$\mathscr{C} = (\mathscr{V} \setminus \mathcal{L}_{\mathscr{M}}) \cup (\mathscr{W} \cap \mathcal{L}_{\mathscr{M}}) = \{\alpha\} \cup \emptyset = \{\alpha\} \tag{3.2.17}$$

The minimum weight is:

$$\omega^* \leftarrow \min_{i \in (\mathscr{V} \cap \mathcal{L}_{\mathscr{M}})} \min_{j \in (\mathscr{W} \setminus \mathcal{L}_{\mathscr{M}})} \{\omega_{ij}\} = \min_{i \in \{\beta,\gamma\}, j \in \{A,B,C\}} \{\omega_{ij}\} = 10 \tag{3.2.18}$$

And the dual functions get updated as:

$$\begin{cases} \kappa(\alpha) = 0 \\ \kappa(\beta) = 10 \\ \kappa(\gamma) = 10 \\ \kappa(A) = 108 \\ \kappa(B) = 125 \\ \kappa(C) = 150 \end{cases} \tag{3.2.19}$$

Going back to evaluating complementary slackness, Tables 3.3 and 3.4 return the new conditions:

	108	125	150
0	108	125	150
10	150	135	175
10	122	148	250

Table 3.3: Complementary slackness $T = 2$

	108	125	150
0	0	0	0
10	32	0	15
10	4	13	90

Table 3.4: Complementary slackness weights $T = 2$

And we go on, iterating, until the optimal matching of size $n = 3$ is found.

The subgraph K_ω is found on Figure 3.5.

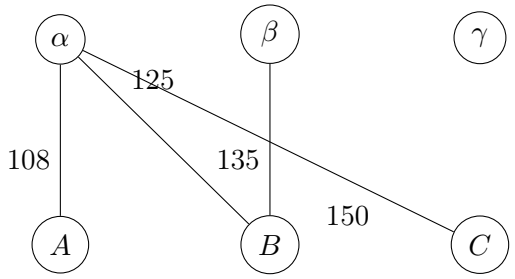


Figure 3.5: Subgraph $T = 2$

A maximum cardinality matching $\mathscr{M} = \{(\alpha, C), (\beta, B)\}$ is of size $|\mathscr{M}| = 2 < n = 3$. It is found using the subroutine of Algorithm 2 and is that of Figure 3.6:

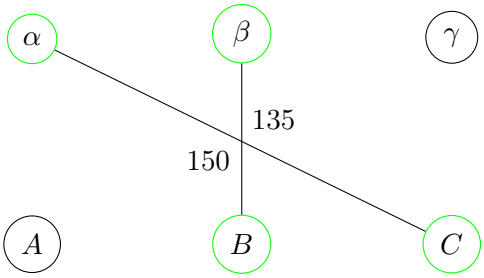


Figure 3.6: Max card $T = 2$

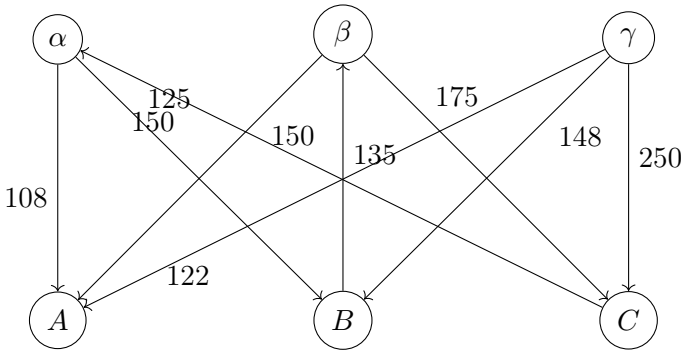


Figure 3.7: Directed graph $T = 2$

Then, its directed path is created in Figure 3.7

Using definition 3.13, the vertices reachable from exposed vertices in Q are

$$\mathcal{L}_{\mathscr{M}} = \{\beta, \gamma, B\} \tag{3.2.20}$$

A cover is built as previously explained as:

$$\mathscr{C} = (\mathscr{V} \setminus \mathcal{L}_{\mathscr{M}}) \cup (\mathscr{W} \cap \mathcal{L}_{\mathscr{M}}) = \{\alpha\} \cup \{B\} = \{\alpha, B\} \tag{3.2.21}$$

The minimum weight is:

$$\omega^* \leftarrow \underset{i \in (\mathscr{V} \cap \mathcal{L}_{\mathscr{M}})}{min} \underset{j \in (\mathscr{W} \setminus \mathcal{L}_{\mathscr{M}})}{} \{\omega_{ij}\} = \underset{i \in \{\alpha,\gamma\}, j \in \{A,C,\}}{min} \{\omega_{ij}\} = 4 \tag{3.2.22}$$

And the dual functions get updated as:

$$\begin{cases} \kappa(\alpha) = 0 \\ \kappa(\beta) = 14 \\ \kappa(\gamma) = 14 \\ \kappa(A) = 108 \\ \kappa(B) = 121 \\ \kappa(C) = 150 \end{cases} \tag{3.2.23}$$

Going back to evaluating complementary slackness, Table 3.5 returns the new conditions:

	108	121	150
0	108	125	150
14	150	135	175
14	122	148	250

Table 3.5: Complementary slackness $T = 3$

	108	121	150
0	0	4	0
14	28	0	11
14	0	13	86

Table 3.6: Complementary slackness weights $T = 3$

Going back to step 3, it now holds that there is an optimal matching in K_ω, shown on Figure 3.8.

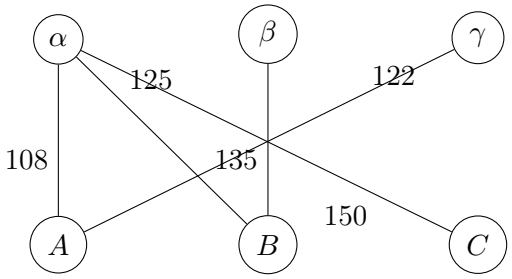


Figure 3.8: Subgraph $T = 3$

A maximum cardinality matching $\mathscr{M} = \{(\alpha, C), (\beta, B), (\gamma, A)\}$ is of size $|\mathscr{M}| = 3 = n$. It is found using the subroutine of Algorithm 2 and is that of Figure 3.9. The procedure has identified an optimal solution.

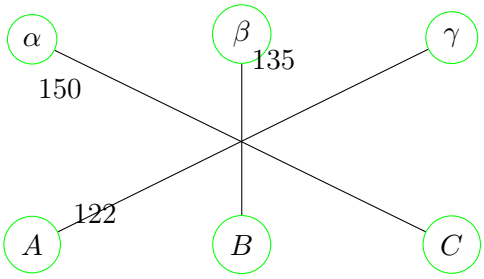


Figure 3.9: Max card $T = 3$, Solution

Building up on relaxations of an integer problem with integral optimality results, a flow fashioned approach guarantees an efficient solution for fairly general problems.

The importance of such method is provided by the omnipresence of the assignment problem in the industrial sector, where scarcity of resources is a constant threat.

The formulation of Definition 1.2 has been largely studied in other settings and contexts. In section 4.1 a generalization is proposed, following the approach of [15]. In section 4.2 the problem of counting the number of matchings in a general graph is considered. The latter is mostly inspired by the work of Moore and some lecture notes by Goemans[16], [17].

Chapter 4

Extensions

4.1 Generalizations of $\mathfrak{P}$

The unique end of science is the honor of the human mind.

Carl Gustav Jacob Jacobi

Without going much into the detail, some elements from group theory will be introduced, to later present important statements.

Definition 4.1 (Total order $\preceq$)**.** A total order is an order (binary) relation $\preceq$ on a set C satisfying $\forall a, b, c \in H$:

- Reflexivity
$$a \preceq a \tag{4.1.1}$$
- transitivity
$$a \preceq b, b \preceq c \implies a \preceq c \tag{4.1.2}$$
- antisymmetry
$$a \preceq b, b \preceq a \implies a = b \tag{4.1.3}$$
- totality
$$a \preceq b \wedge b \preceq a \tag{4.1.4}$$

Definition 4.2 (Totally ordered commutative semigroup $\mathcal{S}$)**.** A totally ordered commutative semigroup is an algebraic structure $(H, \circledast)$ consisting of a set and a binary operation (a semigroup) endowed with a total order $\preceq$.

$$\mathcal{S} \coloneqq (H, \circledast, \preceq) \tag{4.1.5}$$

Assumption 4.3 ($\mathcal{S}$ is a d-monoid)**.** It is further added that $\mathcal{S}$ must be such that $\forall a, b, c \in H$:

- binary operation does not impact order
$$a \preceq b \implies a \circledast c \preceq b \circledast c \tag{4.1.6}$$
- there is a gap-filling element
$$a \preceq b \implies \exists c : a \circledast c = b \tag{4.1.7}$$

A semigroup with this axiomatic requirements is sometimes called a d-monoid [15].

Definition 4.4 (Generalized Assignment Problem $\mathfrak{P}_{gen}$)**.** Given a cost function C two sets, a totally ordered commutative semigroup $\mathcal{S}$ fullfilling Assumption 4.3, a feasible set $\mathscr{S} \subset \mathscr{P} = (\mathscr{A} \times \mathscr{T})^n$ as in Definition 1.3, find the minimum cost feasible solution.
Equivalently in permutation terms:

$$\pi^* = \underset{\pi \in \Pi}{argmin}\Big\{C(\pi)\Big\} \quad C(\pi) = \overset{n}{\underset{i=1}{\circledast}} C_{\pi(i)} = C_{\pi(1)} \circledast C_{\pi(2)} \circledast \ldots \circledast C_{\pi(n)} \tag{4.1.8}$$

Which in matching terms[1] given $\mathscr{M} = \{e_1, \ldots e_n\}$ has cost:

$$C(\mathscr{M}) = C_{e_1} \circledast C_{e_2} \circledast \ldots \circledast C_{e_n} \tag{4.1.9}$$

Both will be used equivalently, as in the end they are the same.

Example 4.5 (d-monoids)**.** The easiest cases are:

- $H = \mathbb{R}, \circledast = +$ and usual order leads a sum assignment problem of finding the extremals of an objective function
- $H = \mathbb{R}, \circledast = max\{\}$ and usual order leads to bottleneck assignment problems, aiming to minimize the latest completion time of agents-tasks assignment.

$$min_{\pi \in \Pi}\Big\{max_{i \in \{1,\ldots,n\}}\{C_{i,\pi(i)}\}\Big\} \tag{4.1.10}$$

The importance of such an approach is encapsulated in the extension of the intuition behind the hungarian algorithm. The idea is to transform the cost matrix through steps that guarantee that the optimal solution will be found.

Definition 4.6 (Admissible transformation $\zeta, \zeta(T)$)**.** An admissible transformation is an operation on the costs that does not impact the relative ordering of feasible solution in $\mathscr{S}$.
A transformation T of the cost C into a new cost $\overline{C}$ is admissible with index $\zeta(T)$ if:

$$C(\mathscr{M}) = \zeta(T) \circledast \overline{C}(\mathscr{M}) \quad \forall \mathscr{M} \in \mathscr{S} \tag{4.1.11}$$

Where a composition of transformations T, S satisfies the trivial identity:

$$C(\mathscr{M}) = \zeta(S) \circledast \zeta(T) \circledast \overline{C}(\mathscr{M}) \quad \forall \mathscr{M} \in \mathscr{S} \tag{4.1.12}$$

Lemma 4.7 (Admissible transformations and optimality)**.** *Let T be admissible with index $\zeta(T)$, and such that there exists a feasible solution $\mathscr{M}^*$ satisfying:*

1. $\overline{C}(\mathscr{M}) \circledast \zeta(T) \preceq \zeta(T) \quad \forall \mathscr{M} \in \mathscr{S}$
2. $\overline{C}(\mathscr{M}^*) \circledast \zeta(T) = \zeta(T)$

Then $\mathscr{M}^*$ is an optimal solution to the problem:

$$\underset{\mathscr{M}}{min}\Big\{ \overset{}{\underset{e \in \mathscr{M}}{\circledast}} C_e \Big\} \tag{4.1.13}$$

With value $\zeta(T)$

Proof. Let $\mathscr{M} \in \mathscr{S}$ be arbitrarily feasible. Using the feasible transformation and the above stated properties:

$$\begin{aligned}
C(\mathscr{M}) &= \zeta(T) \circledast \overline{C}(\mathscr{M}) && \text{By Definition 4.6} && (4.1.14)\\
&\succeq \zeta(T) && \text{by property 1} && (4.1.15)\\
&= \zeta(T) \circledast \overline{C}(\mathscr{M}^*) && \text{by property 2} && (4.1.16)\\
&= \overline{C}(\mathscr{M}^*) && \text{cancelling out } \zeta(T) && (4.1.17)\\
\implies \overline{C}(\mathscr{M}^*) &= \zeta(T) \quad \textit{optimal} && && (4.1.18)
\end{aligned}$$

[1] The notation for a matching is used to avoid confusion, but is to be counted as a feasible solution $\mathscr{M} \in \mathscr{S}$

□

If such a solution does not exist, then it is possible to perform admissible transformations to get closer and closer to an optimal solution, while keeping the cost *valid*. This is a generalization of the dual update of Chapter 3: once a feasible solution does not satisfy the optimality requirement, it is transformed as to get closer to the boundary. The following theorem generalizes in a broader sense the update equations approach of Equations 3.2.7 and 3.2.8, used in Algorithm 3. An easier case proof for the linear assignment is reported in [15], and then stated as a proposition for the general setting. Below the universal case is prosed.

Theorem 4.8 (Admissible transformations for linear assignment problems with general objective)**.** *Express $\mathfrak{P}_{gen}$ as a permutation search. Let $I, J \subseteq \{1, \dots, n\}$, such that $m = |I| + |J| - n \geq 0$ and let:*

$$c^* = min\{C_{ij} | i \in I, j \in J\} \tag{4.1.19}$$

Then, the transformation T defined as:

$$\overline{C}_{ij} \circledast c^* = C_{ij} \quad i \in I, j \in J \tag{4.1.20}$$

$$\overline{C}_{ij} = \begin{cases} C_{ij} \circledast c^* & i \notin I, j \notin J \\ C_{ij} & otherwise \end{cases} \tag{4.1.21}$$

Is admissible with $\zeta(T) = (c)^{\circledast m}$, which is c star operated by itself m times.

Proof. Let $\pi \in \Pi$ be a permutation of $\{1, \dots, n\}$. Moreover define:

- n_0 as the number of $(i, \pi(i))$ pairs where $i \in I, \pi(i) \in J$
- n_1 as the number of $(i, \pi(i))$ pairs where $(i \in I, \pi(i) \notin J) \wedge (i \notin I, \pi(i) \notin J)$
- n_2 as the number of $(i, \pi(i))$ pairs where $i \notin I, \pi(i) \notin J$

It is clear that $n_0 + n_1 + n_2 = n$. Moreover:

$$\begin{aligned} |I| + |J| - n &= 2n_0 + n_1 - n && (4.1.22)\\ &= 2n_0 + n_1 - n_0 - n_1 - n_2 && (4.1.23)\\ &= n_0 - n_2 && (4.1.24)\\ &= m && (4.1.25) \end{aligned}$$

Where equality between Equations 4.1.22, 4.1.25 (assumed) and 4.1.24 will be used. Considering Equation 4.1.25, being independent of the choice of π, it holds $\forall \pi \in \Pi$ [15]. Let $C[i \in I] \coloneqq \circledast_{i \in I} C_{i,\pi(i)}$. Then from Equation 4.1.20:

$$C(\pi) = C[i \in I] + C[i \notin I] \qquad I \subseteq \{1, \dots, n\} \text{ is contained in all the indices} \tag{4.1.26}$$

$$= n_0 \diamond c^* \circledast \overline{C}[i \in I] \circledast C[i \notin I] \quad \text{where } \diamond \text{ is the } \circledast \text{ multiplication} \tag{4.1.27}$$

While from Equation 4.1.21:

$$n_2 \diamond c^* \circledast C[i \notin I] = \overline{C}[i \notin I] \implies C[i \notin I] = C[i \notin I] = \overline{C}[i \notin I] \circledast (-n_2 \diamond c^*) \tag{4.1.28}$$

Thus for an arbitrary permutation $\pi \in \Pi$ it holds that:

$$\begin{aligned} C(\pi) &= n_0 \diamond c^* \circledast \overline{C}[i \in I] \circledast \overline{C}[i \notin I] \circledast (-n_2 \diamond c^*) && (4.1.29)\\ &= (n_0 - n_2) \diamond c^* \circledast \overline{C}(\pi) && (4.1.30)\\ &= m \diamond c^* \circledast \overline{C}(\pi) && (4.1.31)\\ &= (c^*)^{\circledast m} \circledast \overline{C}(\pi) && (4.1.32) \end{aligned}$$

Which implies that:

$$C(\pi) = \zeta(T) \circledast \overline{C}(\pi) : \zeta(T) = (c)^{\circledast m} \quad \forall \pi \in \Pi \tag{4.1.33}$$

□

Drawing from [2], a simplified procedure with these results in hand is outlined in Algorithm 5.

Algorithm 5 Generalized Assignment through admissible transformations

Input: $P \in \mathfrak{P}_{gen}$ as in Definition 4.4

Output: minimum objective function assignment

1: Perform admissible transformations in **Steps 2-5**: ▷ Dual initialization
2: **for** $i \in \{1, \dots, n\}$ **do** ▷ row reductions
   let $I = \{i\}$, $J = \{1, \dots, n\}$, $\zeta = \zeta(T)$ the index
   update $\zeta = \zeta \circledast \zeta(T)$ ▷ Now $\forall j \overline{C}_{ij} \circledast \zeta \succeq \zeta$
3: **end for**
4: **for** $j \in \{1, \dots, n\}$ **do** ▷ column reductions
   let $J = \{j\}$, $I = \{1, \dots, n\}$, $\zeta = \zeta(T)$ the index
   update $\zeta = \zeta \circledast \zeta(T)$ ▷ Now $\exists i : \zeta \circledast \overline{C}_{ij} = \zeta$
5: **end for** ▷ Now $\forall$ row, col $\exists \zeta \circledast \overline{C}_{ij} = \zeta$ generalized zero element
6: Build the subgraph B_ω ▷ made of generalized zero elements
7: Determine a maximum matching $\mathscr{M}$ in B_ω ▷ as before in the optimal edges only
8: **if** $|\mathscr{M}| == n$ **then**
   return $\mathscr{M}$ optimal, ζ optimal cost
9: **end if**
10: determine the cover $\mathscr{C}$ rebuilding I, J as the uncovered rows and columns
11: $\zeta(T) \leftarrow$ admissible transformation on I, J ▷ using Lemma 4.7
12: update $\zeta = \zeta \circledast \zeta(T)$ ▷ Dual update
13: Go back to **Step 6**

A couple remarks on Algorithm 5:

- The dual construction is slightly different, and more similar to finding the minimum for each row or column and subtracting it from the whole row or column. This dual initialization is suggested in many online resources. It is rather straightforward to prove for instances of $\mathfrak{P}$.
- Generalized zero elements are the *zeros* in the tables of previous examples, obtained by enforcing feasible transformations that find the minimum weight element and subtract its value.
- The subgraph matching-finding and cover-finding steps are generalized equivalents of the previous processes, as they are unweighted
- The admissible transformation is found again using Lemma 4.7 on Step 12

Observation 4.9 (On the number of admissible transformations)**.** *The times $\zeta(T)$ is applied to find an optimal solution to the algebraic assignment problem are bounded by $n^2 - 2n - 3$, as by [2].*

In [2],[18] it is also claimed that, adding a weak cancellation rule of the form:

$$a \circledast c = b \circledast c \implies \Big(a = b\Big) \vee \Big(a \circledast c = b\Big) \tag{4.1.34}$$

The complexity of such procedure is potentially improved to:

$$T(n) \in O(n^3) \tag{4.1.35}$$

4.2 Counting Hardness

As for everything else, so for a mathematical theory: beauty can be perceived but not explained.

Arthur Cayley

Motivated by the interesting question of counting the number of perfect matchings for a graph $B \in \mathscr{B}$, what follows is the result of a study on Chapters 13-14 of [17], enlarged by some lecture notes [16], which propose a framework for both bipartite graphs and general graphs with specific properties.

Definition 4.10 (Symmetric Group S_n)**.** Given a set $\{1, \dots, n\}$ and an operation $\circ$ of function composition, a symmetric group is the set of all bijections of the set $\{1, \dots, n\}$.

$$S_n \coloneqq \Big\{ \pi : \{1, \dots, n\} \to \{1, \dots, n\},\ \circ \Big\} \tag{4.2.1}$$

Definition 4.11 (Permanent $per(\cdot)$)**.** Given a matrix R, its permanent is defined as:

$$per(R) \coloneqq \sum_{\pi \in S_n} \prod_{i=1}^{n} R_{i,\pi(i)} \tag{4.2.2}$$

The permanent is a cousin of the determinant, which looks similar but has different properties and meaning. Below an example shows how to calculate it for square 2×2 matrices.

Example 4.12 (Permanent of 2×2 matrix)**.** Given a matrix $R = \begin{bmatrix} r_{11} & r_{12} \\ r_{21} & r_{22} \end{bmatrix} \in \mathfrak{M}_{2,2}$ we have trivially $n = 2$, thus the group is composed of permutations of the set $\{1, 2\}$ giving:

$$S_2 = \Big\{ \{1, 2\}, \{2, 1\},\ \circ \Big\}$$

Where there are two permutations π_1, π_2. The first is the identity, the second just inverts the the numbers $\pi_2(1) = 2, \pi_2(2) = 1$. The indices are $i = 1, i = 2$, this means that:

$$per(R) = (r_{11} \cdot r_{22}) + (r_{12} \cdot r_{21})$$

Which looks like a 2×2 determinant ignoring the sign in the second term.

Valiant [19] proposed a characterization of counting problems into a new complexity class.

Definition 4.13 ($\#P$ complexity class)**.** $\#P$ comprises all counting problems with a polynomial time output verifier. Any problem asking the number of solutions of a specific instance can be included in this class.

Assumption 4.14 (Counting problems notation)**.** For a counting problem of evaluating the number of optimal (max or min size solutions) the notation will be:

$$\#problem$$

Example 4.15 (A problem in $\#P$)**.** Counting the number of permutations of a set $\{1, \dots, n\}$ which start with number n is a $\#P$ problem. The question indeed is finding how many satisfactory solutions exist given an instance (a set in this case).

Following the classic computational complexity framework, also *#P-complete* and *#P-hard* classes can be considered. Indeed, the first problem added to the *#P-complete* class is that of computing the permanent of a $\{0,1\}$ matrix.

Theorem 4.16 (Valiant Theorem)**.** *Given a $\{0,1\}$ matrix R computing its permanent is #P-complete.*

Sketch from [17]. It can be argued that:

$$\#3\text{-}SAT \leq permanent \leq positive\ permanent \leq 0\text{-}1\ permanent \tag{4.2.3}$$

□

The above result is crucial to understand the underlying complexity of such problem. Using an adjacency matrix, it is possible to represent available connections in any instance of $\mathfrak{G}$. Having this intuitive result in mind, one could conclude that computing the permanent of a graph G is somewhat expensive computationally. However, there is more to this, as the permanent of a graph is exactly the number of perfect matchings in a graph $G \in \mathfrak{G}$.

Theorem 4.17 (Number of perfect matchings and permanent)**.** *Given a bipartite graph $B \in \mathscr{B}$ with equal weights, there exists a $\{0,1\}$ matrix R such that its permanent is equal to the number of perfect matchings of B. Namely:*

$$\implies \exists R \in \mathfrak{M}_{n,n}[0,1] \ : \ |\{\mathscr{M}_1 \ldots,\}| = per(R) \ : \forall \mathscr{M} \ |\mathscr{M}| = \nu(B) \tag{4.2.4}$$

Proof. Define R as:

$$R := \begin{cases} R_{ij} = 1 & if\ \exists e = (v_i, w_j) \in \mathscr{E} \\ R_{ij} = 0 & otherwise \end{cases} \tag{4.2.5}$$

R is the square root of an adjacency matrix[2] A, as by the graph being bipartite it holds that:

$$A = \begin{bmatrix} 0_{n,n} & R \\ R^T & 0_{n,n} \end{bmatrix} \in \mathfrak{M}_{2n,2n} \ \ R \in \mathfrak{M}_{n,n} \tag{4.2.6}$$

For a matching $\mathscr{M}_q$ we have that it consists of a permutation π_q where v_i is matched to w_j. Thus $\pi_q \in S_n$ is a permutation where in $per(R)$ the term is:

$$\prod_{i=1}^{n} R_{i,\pi_q(i)} = 1 \tag{4.2.7}$$

On the other side of the equivalence, considering the set of permutations $\pi_q \in S_n$ such that $\prod_{i=1}^{n} R_{i,\pi_q(i)} = 1$, each of them corresponds to a unique perfect matching identified by π_q.
Since R is a $\{0,1\}$ matrix, each term in the sum of all $\pi \in S_n$ is:

- 0 if $\exists v_i \in \mathscr{V} \ : \ \pi(i) = j \ where \nexists e = (v_i, w_j) \in \mathscr{E}$
- 1 otherwise as it is a valid perfect matching with all existing edges.

Thus each term in the outer sum is either 0 or 1 and when it is one it corresponds uniquely to a perfect matching. □

[2]This is **NOT** the totally unimodular matrix. To avoid sloppy notation the A symbol is used again. The topics do not relate in this report.

Corollary 4.18 (Number of solutions and permanent with weights)**.** *Given a bipartite graph $B \in \mathscr{B}$ with weighted edges, assuming the weight of a matching $\mathscr{M}$ is the product of the weights of its edges and is denoted as $C(\mathscr{M})$, the permanent is the sum of the weights of all perfect matchings.*

$$if\ C(\mathscr{M}) = \prod_{e_i \in \mathscr{M}} e_i \forall \mathscr{M} \implies per(R) = \sum_{\mathscr{M}} C(\mathscr{M}) \tag{4.2.8}$$

Proof. Everything follows Theorem 4.17, apart from the product, which now is the product of the weights of the edges. □

From the results of Theorem 4.16 and 4.17 it can be argued that $\#\mathscr{M}$ is *#P-complete*.

Remark. Recalling Equation 4.2.6 it is worth noticing that:

$$per(A) = perm(R)^2 \tag{4.2.9}$$

In some cases, it might be easier to work with the adjacency matrix.

This fact strengthens even more the innate challenge that solving $\mathfrak{P}$ is. While finding one solution might in some bipartite instances be feasible, as will be shown in the next sections, this is not an easy to answer question in general. Nevertheless, for what concerns finding one optimal solution, the hungarian algorithm of section 3 is a viable option.

Remark (On multiple vs a single optimal)**.** The hardness of counting the number of perfect matchings does not influence the hardness of finding only one. Viceversa, if a method to identify an extremal can be formulated, this does not in turn mean that counting is not *#P-complete*.

A useful example of efficiently solvable problem is brought to the attention of the reader.

As a first step, the easy determinant & spanning trees problem is presented, for a more formal treatment, it is again possible to consult [17].

Definition 4.19 (Determinant $det(\cdot)$)**.** Given a matrix $M \in \mathfrak{M}_{n,n}$ its determinant is defined as:

$$det(M) \coloneqq \sum_{\pi \in S_n} (-1)^{sgn(\pi)} \prod_{i=1}^{n} M_{i,\pi(i)} \tag{4.2.10}$$

It differs from the permanent of Definition 4.11 only by $(-1)^{sgn(\pi)}$ where sgn counts the number of transpositions of two elements in the given permutation (the **parity**) to get the identity permutation.

Definition 4.20 (Spanning Tree $\mathcal{T}$)**.** Given a graph $G \in \mathfrak{G} : |\mathscr{V}| = n$ a spanning tree $\mathcal{T}$ is a connected acyclyc subgraph $\mathcal{T} \subseteq G$ containing all of the vertices in $\mathscr{V}$.

Definition 4.21 (Laplacian L)**.** Given a graph $G \in \mathfrak{G}\ : |\mathscr{V}| = n$ its Laplacian is defined as:

$$L \in \mathfrak{M}_{n,n} \quad L_{ij} = \begin{cases} d_i & if\ i = j \\ -1 & if \exists e = (i,j) \in \mathscr{E} \\ 0 & otherwise \end{cases} \tag{4.2.11}$$

A famous result by Kirchhoff links the above Definitions from a computational perspective.

Theorem 4.22 (Kirchhoff's Matrix-Tree Theorem)**.** *Given a graph $G \in \mathfrak{G}$ denote by $\#\mathcal{T}$ the number of spanning trees. Then:*

$$\#\mathcal{T} = det(L^{(-ii)}) \tag{4.2.12}$$

Where $L^{(-ii)}$ is the matrix L with row i and column i removed.

Proof. Found in [17]. □

Theorem 4.22 shows that counting spanning trees and computing the determinant are related. The Laplacian L can be computed in polynomial time from the adjacency matrix, and the determinant of a matrix is also polynomially computable using the method of *Gaussian Elimination*.

Proposition 4.23 (Determinant is in $\#P$)**.** *Computing the determinant of a matrix, and thus counting the number of spanning trees $\mathcal{T}$ for a graph $G \in \mathfrak{G}$ is in $\#P$.*

In some special cases, efficiently computing the determinant could be helpful to calculate the permanent. In practical terms $per(\cdot)$ and $det(\cdot)$ differ only by the $(-1)^{sgn(\pi)}$ term. For a bipartite graph, thanks to some adjustments, the aim is to show that:

$$\exists R^\star : per(R) = det(R^\star) \tag{4.2.13}$$

For the purpose of showing where this condition can be proved, it is first of all worth stating that a balanced (not necessarily complete) bipartite graph $B = \{(\mathscr{V} \cup \mathscr{W}), \mathscr{E}\} \in \mathscr{B} : |\mathscr{V}| = |\mathscr{W}| = n\}$ will be considered.

A perfect matching in B can be expressed as a permutation π, and the permanent of the *root* adjacency matrix R, by Theorem 4.17 counts how many of these are possible, while the determinant is of the form:

$$det(R) = \sum_{\pi\ perfect} (-1)^{sgn(\pi)} \tag{4.2.14}$$

The computational trick avoids the considering the (-1) power by adding weights that *align* the orientation of the permutations.

Consider a graph B as before where R is consequently:

$$R = \begin{cases} R_{ij} = 1 & if\ \exists e = (i,j) \in \mathscr{E},\ i \in \mathscr{V},\ j \in \mathscr{W} \\ 0 & otherwise \end{cases} \tag{4.2.15}$$

Then, adding weights $w_{ij} = \pm 1$ let $R^\star_{ij} \coloneqq R_{ij} w_{ij}$. It holds that each matching in the determinant matrix computation has weight:

$$w(\pi) = \prod_{i=1}^{n} w_{i,\pi(i)} \tag{4.2.16}$$

$$\implies det(R^\star) = \sum_{\pi\ perfect} (-1)^{sgn(\pi)} w(\pi) \tag{4.2.17}$$

If it were possible to devise a procedure such that the weights return the same sign for each permutation, then it would be the case that:

$$|det(R^\star)| = per(R) = \#\mathcal{T} = \#\mathscr{M} \tag{4.2.18}$$

Or, in adjacency matrix terms:

$$A^\star = \begin{bmatrix} 0 & R^\star \\ R^{\star T} & 0 \end{bmatrix} = \begin{bmatrix} R^\star & 0 \\ 0 & R^{\star T} \end{bmatrix} \begin{bmatrix} 0 & I_n \\ I_n & 0 \end{bmatrix} \tag{4.2.19}$$

$$\implies det(A^\star) = det(R^\star) det(R^{\star T}) det\left(\begin{bmatrix} 0 & I_n \\ I_n & 0 \end{bmatrix} \right) \tag{4.2.20}$$

Where the determinant of the antidiagonal identity matrix can be evaluated using the definition of determinant, and noting that it depends only on the parity of the permutation.

Informal
If there are n even elements, then $sgn(\pi) = +1$, else if n is odd, then $sgn(\pi) = -1$. As a matter of fact, the antidiagonal identity matrix swaps the elements in reversed order. To bring the matrix to the identity, the number of *moves* needed is

$$n-1+n-2+\ldots = \sum_{k=1}^{n-1} = \frac{n(n-1)}{2} \implies (-1)^{\frac{(n-1)(n-2)}{2}} = (-1)^n$$

Which implies that:

$$det\left(\begin{bmatrix} 0 & I_n \\ I_n & 0 \end{bmatrix} \right) = (-1)^n det(I_{2n}) = (-1)^n \tag{4.2.21}$$

Coming back to the calculation:

$$\implies det(A^\star) = det(R^\star)^2 (-1)^n \tag{4.2.22}$$

$$\implies |det(A^\star)| = per(R^\star)^2 = (\#\mathscr{M})^2 \tag{4.2.23}$$

The peculiarity of matchings in bipartite graphs can be extended to general graphs after introducing cycle covers. This is a pivotal step that allows to link the determinant and the adjacency matrix in specific cases.

Definition 4.24 (Cycle Cover $\mathcal{C}^\circlearrowleft$)**.** Given a graph $G \in \mathfrak{G}$ a cycle cover is a set of disjoint directed paths that span $\mathscr{V}$.

Example 4.25 (Cycle cover for simple graph)**.** A cycle cover for a three node graph is shown in Figure 4.1.

Lemma 4.26 (Permanent and counting cycle covers)**.** *Given a graph $G \in \mathfrak{G}$ with adjacency matrix A it holds that:*

$$per(A) = \#\mathcal{C}^\circlearrowleft \tag{4.2.24}$$

Proof. Consider a cycle cover. It can be interpreted as a permutation where the arrows in the cycle indicate where the element maps from and where it maps to.
Consider a permutation, it maps uniquely vertices to other vertices without repetition, it is thus a cycle cover.
Recalling Definition 4.11, it holds that the permanent of the adjacency matrix is such that each term is a vertex cover.

$$per(A) = \#\mathcal{C}^\circlearrowleft \tag{4.2.25}$$

□

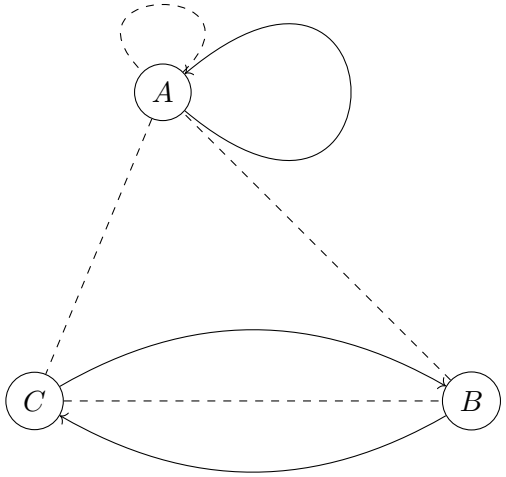


Figure 4.1: Cycle cover (full) simple graph (dashed)

For bipartite graphs, an easy statement is the following.

Theorem 4.27 (Bipartite cycle cover and matchings)**.** *For a bipartite graph $B \in \mathscr{B}$ it holds that:*

$$\#\mathcal{C}^{\circlearrowleft} = (\#\mathscr{M})^2 \tag{4.2.26}$$

Proof. The convention presented to support this proof is the following:
For the graph B, distinguish elements from the two vertex sets by setting $v \in \mathscr{V}$ black and $w \in \mathscr{W}$ white. Given two matchings $\mathscr{M}_1, \mathscr{M}_2$, obtain a cycle cover as:

$$\mathcal{C}^{\circlearrowleft} = \pi : \begin{cases} i \to \pi(i), & from\ \mathscr{M}_1 \implies i\ black \\ i \to \pi(i), & from\ \mathscr{M}_2 \implies i\ white \end{cases} \tag{4.2.27}$$

Now it is possible to prove the claim.
(from matchings) Given two matchings, consider their symmetric difference $\mathscr{M}_1 \Delta \mathscr{M}_2$, it holds that edges appear in one of the two but not in both. For all of those, orient them as the convention suggests. For all shared edges in $\mathscr{M}_1 \cap \mathscr{M}_2$ instead, add edges going back and forth from $i \to \pi(i), \pi(i) \to i$ as it is the case that $i = \pi(i)$.
The result is a cycle cover. If $\#\mathscr{M} = k$ then the elements can be combined in k^2 pairs (a table), and the number of cycle covers is k^2.
(from covers) Conversely, consider a set of k even cycles, with isolated edges potentially, covering a graph $B \in \mathscr{B}$. Denote it as σ. Each cycle has a clockwise and a counterclockwise direction. Thus there are 2^k cycle covers, where the orientation identifies a direction (either $\mathscr{V} \to \mathscr{W}$ or the opposite), and thus a positioning in either of the two matchings. Therefore, there are 2^k pairs $\mathscr{M}_1, \mathscr{M}_2$ such that $\mathscr{M}_1 \Delta \mathscr{M}_2 = \pi$. If there are 2^k pairs, then the number of matchings is necessarily k^2, by a simple combinatoric argument. □

Counting cycle covers and matchings is hard, but there are problems which are not hard at all. In some instances, it will be possible to state whether they provide an exact solution to the harder ones. What follows is a path towards this result.

4.2.1 Pfaffian Orientations

In some cases, specific properties of the graph chosen allow to simplify the calculation of the permanent.

A modification of the adjacency matrix A denoted as $A^\star$, crafted as to make odd length cycles null, would suffice to make the determinant of $A^\star$ count only useful covers. Let $A^\star$ be antisymmetric (such that $A^\star = -A^{\star T}$).

Following the path proposed in [16] consider a general graph $G = \{\mathscr{V}, \mathscr{E}\} \in \mathfrak{G}$ where $|\mathscr{V}| = n$. The custom matrix $A^\star$ is built as:

$$A^\star = \begin{cases} A^\star_{ij} = -A^\star_{ij} & if\ e = (i,j) \in \vec{\mathscr{E}} \\ A^\star_{ij} = 0 & otherwise \end{cases} \tag{4.2.28}$$

Where the sign of the edges is determined by a custom direction assigned. The graph induced by $A^\star$ is oriented by the edges $\vec{\mathscr{E}}$.

Observation 4.28 (On the custom matrix $A^\star$)**.** *$A^\star$ is antisymmetric, and is such that:*

$$\begin{aligned} &det(A^\star) = det(-A^{\star T}) = (-1)^n det(A^\star) \\ \implies &if\ n\ odd \implies det(A^\star) = -det(A^\star) \implies det(A^\star) = 0 \end{aligned}$$

Definition 4.29 (Odd and even cycles)**.** A cycle is even if it is of even length. A cycle cover is odd if it contains at least one cycle of odd length.

Considering a permutation on $S_n : n\ even$ it is possible to identify the perfect matching represented as:

$$\mathscr{M} = \{(\pi(2i-1), \pi(2i)) : 1 \leq i \leq \frac{n}{2}\} \tag{4.2.29}$$

Some useful facts will be reported in the next Lemma.

Lemma 4.30 (Some identities in permutations, cycles, covers)**.** *The following facts will be used later in the section:*

- $\pi = \mathscr{M}_1 \Delta \mathscr{M}_2 \implies \pi$ *is a cycle*
- $\mathscr{M}$ *identifies a permutation* $\pi \in S_n$ *as in Equation 4.2.29*
- *The number of cycle covers is even, and arises from cycles* $c = \mathscr{M}\Delta\mathscr{M}'$*. Then, the number of elements in such cycle is even*
- $c = \mathscr{M}\Delta\mathscr{M}' \implies \pi' = \pi \circ c \wedge sgn(\pi') = -sgn(\pi)$

For a given matching, many permutations correspond to it. To find a useful orientation, the weight of a matching is defined below:

Definition 4.31 (Matching weight $w(\mathscr{M})$)**.** Consider $\pi \in S_n$ with associated $\mathscr{M}$ and $A^\star$ antisymmetric. The weight of a matching is defined as:

$$w(\mathscr{M}) = sgn(\pi) \prod_{i=1}^{\frac{n}{2}} A^\star_{\pi(2i-1),\pi(2i)} \tag{4.2.30}$$

The surjectivity of the $\pi \to \mathscr{M}$ map does not impact $w(\mathscr{M})$. Indeed, the object is well defined. To prove this consider π_2 and π_1 where both map to $\mathscr{M}$. Through a sequence of elementary moves chosen from the below list one will be reduced to the other without changing the weight of the matching.

1. change the order of an edge: switch $\pi(2k-1)$ and $\pi(2k)$ for some k
2. exchange two edges: switch $\pi(2k-1), \pi(2k)$ with $\pi(2j-1), \pi(2j)$ for some j, k

For item 1, the sign of the permutation changes, but the A_{ij} change as well, leaving $w(\mathscr{M})$ unchanged.
For item 2 there is no parity change as the edges just switch completely.

While in a bipartite graph cycle covers have always odd length, this does not hold in general for any graph $G \in \mathfrak{G}$. Indeed, the following Lemma generalizes the result of Theorem 4.27.

Lemma 4.32 (Even cycle covers and perfect matchings)**.** *For a graph $G \in \mathfrak{G}$ there exists a bijection between cycle covers with only even cycles and perfect matchings. Thus the number of perfect matchings squared is equal to the number of even cycle covers. Namely:*

$$\exists F : \xi_n \to \{\mathscr{M}\} \times \{\mathscr{M}\} \quad : F \text{ bijective where } \xi_n = \{\pi \in S_n : \text{only even cycles}\} \tag{4.2.31}$$

$$\implies \#\mathcal{C}^{\circlearrowleft}_{\text{all even cycles}} = (\#\mathscr{M})^2 \tag{4.2.32}$$

Proof. Apply the reasoning of Theorem 4.27, considering that a cycle of odd length does not identify a matching as it does not relate to specific edges.
For each $\pi \in \xi_n$ consider the smallest index u and let $(u, \pi(u)) \in \mathscr{M}$, similarly $(\pi(u), \pi(\pi(u))) \in \mathscr{M}'$. This can be done in alternation as every cycle is even, and the split does not have any element of the permutations left out.
Consider the union of two matchings $\mathscr{M} \cup \mathscr{M}'$. This is a set of cycles. For each cycle, orient it such that the lowest index u has an outgoing connection. This procedure reproduces the permutation π.
Then, the map $F(\pi) = (\mathscr{M}, \mathscr{M}')$ is a bijection, and it is possible to state that:

$$\#\mathcal{C}^{\circlearrowleft}_{\text{all even cycles}} = |\xi_n| = (|\{\mathscr{M}\}|)^2 = (\#\mathscr{M})^2 \tag{4.2.33}$$

□

Lemma 4.32 is then used to prove a famous result by Cayley.

Definition 4.33 (Pfaffian $Pf(\cdot)$)**.** Given $A^\star \in \mathfrak{M}_{n,n}$ antisymmetric with n even, the Pfaffian of $A^\star$ is the sum of the weights of perfect matchings:

$$Pf(A^\star) \coloneqq \sum_{\mathscr{M}\ perfect} w(\mathscr{M}) \tag{4.2.34}$$

Theorem 4.34 (Cayley's Theorem)**.** *Let $A^\star \in \mathfrak{M}_{n,n}$ be antisymmetric, where n is even. Then:*

$$det(A^\star) = Pf(A^\star)^2 \tag{4.2.35}$$

Proof. From Definition 4.19:

$$det(A^\star) = \sum_{\pi \in S_n} (-1)^{sgn(\pi)} \prod_{i=1}^{n} A^\star_{i,\pi(i)} \tag{4.2.36}$$

The following are two useful observations:

1. By the fact that $A^\star$ is antisymmetric the diagonal elements are null $A^\star_{ii} = 0 \forall i$. Then, permutations mapping vertices to the same vertex cancel out in the sum:

$$\forall \pi \in S_n | \exists i^* : i^* \to \pi(i^*) = i^* \implies \prod_{i=1}^{n} A^\star_{i,\pi(i)} = 0 \tag{4.2.37}$$

2. Considering $\pi \in S_n \setminus \xi_n$ there is always at least one odd cycle in π. Consider the odd cycle C^* containing the smallest index. Reverse C^*. A new permutation is obtained, where:
$$\pi^{new} = \pi \wedge sgn(\pi^{new}) = sgn(\pi) \tag{4.2.38}$$
However, by the antisymmetry of $A^\star$, and the oddness of the cycle, the entries invert their sign, but the number inverted of signs is odd, and the result of the multiplication of entries is inverted in sign in the new permutation:
$$\implies \prod_{i \in C^*} A^\star_{i,\pi(i)} = -\prod_{i \in C^*} A^\star_{i,\pi^{new}(i)} \tag{4.2.39}$$
Thus, for each permutation with an odd cycle inside, the determinant gets canceled out by another permutation.

From point 1 permutations with even cycles that map vertices to themselves are ignored, from point 2 permutations with odd cycles are ignored. Then, the determinant is a sum over permutations with only even cycles $\pi = (c_1) \circ (c_2) \circ \ldots \circ (c_k) : k \; even.$

$$\implies det(A^\star) = \sum_{\pi \in S_n} (-1)^{sgn(\pi)} \prod_{i=1}^{n} A^\star_{i,\pi(i)} \tag{4.2.40}$$

For the Pfaffian, it holds that:

$$Pf(A^\star)^2 = \sum_{\mathscr{M} \; perfect} w(\mathscr{M}) \sum_{\mathscr{M}' \; perfect} w(\mathscr{M}') \tag{4.2.41}$$
$$= \sum_{\mathscr{M} \times \mathscr{M}' \; perfect} w(\mathscr{M}) w(\mathscr{M}') \tag{4.2.42}$$

Which is a sum of multiplications. Using Lemma 4.32, the map $F(\pi) = (\mathscr{M}, \mathscr{M}')$ is a bijection, with $\pi \in \xi_n$ made of k even cycles. To generate the weight of of the matching $\mathscr{M}$ from π, use Definition 4.31, recording elements of each cycle starting from their lowest index. It holds that it is represented through a permutation α where:

$$w(\mathscr{M}) = sgn(\alpha) \prod_{i=1}^{\frac{n}{2}} A^\star_{\alpha(2i-1),\alpha(2i)} \tag{4.2.43}$$

And similarly for $\mathscr{M}'$ there will be an α' representation, obtained by composing with each cycle of π. Namely:

$$\alpha \circ c_i \forall i \in \{1, \ldots, k\} : \pi = (c_1) \circ (c_2) \ldots \circ (c_k) \tag{4.2.44}$$

Therefore, $sgn(\alpha') = sgn(\alpha)(-1)^k : k\ even$ where k is the number of cycles in α. Calculating the product of the two weights:

$$w(\mathscr{M})w(\mathscr{M}') = sgn(\alpha)\prod_{i=1}^{\frac{n}{2}} A^{\star}_{\alpha(2i-1),\alpha(2i)} sgn(\alpha')\prod_{i=1}^{\frac{n}{2}} A^{\star}_{\alpha'(2i-1),\alpha'(2i)} \quad \text{By Equation 4.2.43} \tag{4.2.45}$$

$$= sgn(\alpha)sgn(\alpha')\prod_{i=1}^{\frac{n}{2}} A^{\star}_{\alpha(2i-1),\alpha(2i)} A^{\star}_{\alpha'(2i-1),\alpha'(2i)} \quad \text{collect same index} \tag{4.2.46}$$

$$= sgn(\alpha)^2(-1)^k\prod_{i=1}^{\frac{n}{2}} A^{\star}_{\alpha(2i-1),\alpha(2i)} A^{\star}_{\alpha'(2i-1),\alpha'(2i)} \quad \text{by the identity above} \tag{4.2.47}$$

$$= (-1)^k\prod_{i=1}^{\frac{n}{2}} A^{\star}_{\alpha(2i-1),\alpha(2i)} A^{\star}_{\alpha'(2i-1),\alpha'(2i)} \quad \text{by } sgn(\alpha)^2 = 1 \tag{4.2.48}$$

$$= (-1)^k\prod_{i=1}^{n} A^{\star}_{i,\pi(i)} \quad \text{By Lemma 4.32} \tag{4.2.49}$$

Using the fact that k is even in π, it is the case that $sgn(\pi) = (-1)^k$. Together with this, using Equations 4.2.40, 4.2.42 and 4.2.49 altogether:

$$Pf(A^{\star})^2 = \sum_{\mathscr{M}\times\mathscr{M}'\ perfect} w(\mathscr{M})w(\mathscr{M}') \quad \text{Eq. 4.2.42} \tag{4.2.50}$$

$$= \sum_{\pi\in S_n}(-1)^k\prod_{i=1}^{n} A^{\star}_{i,\pi(i)} \quad \text{Eq. 4.2.49} \tag{4.2.51}$$

$$= \sum_{\pi\in S_n} sgn(\pi)\prod_{i=1}^{n} A^{\star}_{i,\pi(i)} \quad \text{identity above paragraph} \tag{4.2.52}$$

$$= det(A^{\star}) \quad \text{Eq. 4.2.40} \tag{4.2.53}$$

And the claim is proved. □

This result holds for general antisymmetric matrices, but considering a *unitary* entry $A^{\star}$, something can be added in this direction. If $A^{\star}$ is as:

$$A^{\star} := \begin{cases} A^{\star}_{ij} = +1 & if\ i \to j \\ A^{\star}_{ij} = -1 & if\ j \to i \\ A^{\star}_{ij} = 0 & otherwise \end{cases} \tag{4.2.54}$$

Then, for each matching $w(\mathscr{M}) = \pm 1$, and making all weights positive (or negative), it could be possible to evaluate the determinant through the Pfaffian as:

$$Pf(A^{\star})^2 = det(A^{\star}) = \#\mathcal{C}^{\circlearrowleft}_{all\ even} = (\#\mathscr{M})^2 \tag{4.2.55}$$

This is the only requirement of a Pfaffian orientation:

Definition 4.35 (Pfaffian orientation)**.** Given $G \in \mathfrak{G}$ a Pfaffian orientation $\vec{\mathscr{E}}$ is such that for all perfect matchings the weight is the same.

$$\vec{\mathscr{E}} : \forall \mathscr{M}, \mathscr{M}'\ perfect \quad w(\mathscr{M}) = w(\mathscr{M}') \tag{4.2.56}$$

Corollary 4.36 (Pfaffian orientation and determinant)**.** *Let $G \in \mathfrak{G}$. Assume G admits a Pfaffian orientation $\vec{\mathscr{E}}$. Build $A^\star$ as in Equation 4.2.54. Then $A^\star$ is antisymmetric and:*

$$\#\mathscr{M} = \sqrt{det(A^\star)} \tag{4.2.57}$$

Proof. $A^\star$ as in Equation 4.2.54 is necessarily antisymmetric. $A^\star$ is also of even size: $A^\star \in \mathfrak{M}_{n,n}$, where $\frac{n}{2}$ is the size of $\mathscr{V}$. Using Theorem 4.34, the Pfaffian orientation property, unitary weights, and Lemma 4.32 it is possible to state that:

$$\implies \sqrt{det(A^\star)} = Pf(A^\star) \qquad \text{Theorem 4.34} \tag{4.2.58}$$

$$= \sqrt{\#\mathcal{C}^{\circlearrowleft}_{all\ even}} \qquad \text{Pfaffian orientation Def. 4.35, unitary weights} \tag{4.2.59}$$

$$= (\#\mathscr{M}) \qquad \text{Lemma 4.32} \tag{4.2.60}$$

□

Having as a purpose that of finding a graph which admits a Pfaffian orientation, it is worth stressing again that this does not imply that *#-P complete* problems such as computing the permanent are made efficient. As a matter of fact, not all graphs in $\mathfrak{G}$ admit a Pfaffian orientation. One condition, which will be useful to prove the next Theorem, is presented in the following Lemma[3].

Lemma 4.37 (Cycles, perfect matchings, backward oddness)**.** *Let $G \in \mathfrak{G}$. An orientation $\vec{\mathscr{E}}$ is Pfaffian if for every cycle c such that the remaining vertices admit a perfect matching has a number of backward $j \to i$ edges which is **odd**. Denoting $V[c]$ as the vertices encountered along c:*

$$\vec{\mathscr{E}}\ where \forall c\ cycles | \exists \mathscr{M}\ perfect\ in\ \mathscr{V} \setminus V[c] : |j \to i|\ in\ c\ is\ odd \implies \vec{E}\ Pfaffian \tag{4.2.61}$$

Proof. Perfect matchings in the Pfaffian orientation are such that :

$$\mathscr{M} \Delta \mathscr{M}' = c \tag{4.2.62}$$

Where c is a cycle such that:

1. it has even size by Lemma 4.32
2. G admits a perfect matching in $\mathscr{V} \setminus V[c]$ [16]

Thus, a Pfaffian orientation admits a cycle of this form.
Any cycle satisfying points 1 and 2 is the result of the symmetric difference of two perfect matchings. Using $A^\star$ we have that $w(\mathscr{M})w(\mathscr{M}') = \pm 1$ and that for the orientation to be Pfaffian it must additionally hold that $w(\mathscr{M})w(\mathscr{M}') = 1$. Thus, the difference in sign between $w(\mathscr{M})$ and $w(\mathscr{M}')$ can be evaluated considering their generators α and α' where $\alpha' = \alpha \circ c$. Indeed:

$$\implies sgn(\alpha') = -sgn(\alpha) \implies w(\mathscr{M})w(\mathscr{M}') = sgn(\alpha)sgn(\alpha')(-1)^k \tag{4.2.63}$$

$$= sgn(\alpha)^2(-1)(-1)^k \tag{4.2.64}$$

$$= (-1)(-1)^k \tag{4.2.65}$$

Where k is the number of backward edges in c. For the orientation to be pfaffian, it is necessary that:

$$w(\mathscr{M})w(\mathscr{M}') = 1 \implies (-1)(-1)^k = 1 \implies k\ odd \tag{4.2.66}$$

[3] With a very bad title!

□

Observation 4.38 (An easy implication of Lemma 4.37)**.** *All cycles must be of even length. If the number of backward cycles is odd, then the number of forward cycles is odd as well.*

The result just proved holds for planar graphs, defined below, as shown through a stronger statement.

Definition 4.39 (Planar Graph)**.** A planar graph is a graph $G \in \mathfrak{G}$ that can be embedded in an $\mathbb{R}^2$ space with no crossing edges. By construction, it divides the space into regions, which are called *faces*. Also the region outside the graph is an *external face*.

A famous result by Kasteleyn states that planar graphs always admit a Pfaffian orientation. To prove this, a peculiarity of planar graphs will be used.

Lemma 4.40 (Forward clockwise edges and internal vertices oddness)**.** *Let $G \in \mathfrak{G}$ be planar. Then, for any cycle c the sum of the number of forward clockwise edges f + number of vertices k internal to c is odd*

$$(f + k)(mod 2) = 1 \tag{4.2.67}$$

Proof. By induction on k, considering a cycle c.
Base case: for $k = 0$ it holds by assumption. In the above theorem it is assumed that f is odd.

$$(f)(mod 2) = 1 \tag{4.2.68}$$

Induction Hypothesis: assume $k > 0$ so that c is a cycle around more than one face.
Conclusion: Consider a cycle c enclosing two faces, where $(f_1 + k_1)(mod 2) = 1$ and $(f_2 + k_2)(mod 2) = 1$ by assumption. The faces that c cycles around of partition the space, and c is basically the union of the two cycles $c = c_1 \cup c_2$ ignoring its interior. Taking into account the path shared between c_1 and c_2, denote the number of internal vertices of the path as b.
It can be noticed that:

$$k = k_1 + k_2 + b \qquad \text{internal + internal + internal shared} \tag{4.2.69}$$

$$f = f_1 + f_2 - (b - 1) \qquad \text{edges + edges - directions invert} \tag{4.2.70}$$

Where in the second equality the subtraction is to avoid inserting edges of the internal shared path.
Then, the number of internal vertices plus the number of forward clockwise edges is:

$$k + f = k_1 + k_2 + b + f_1 + f_2 - b + 1 \tag{4.2.71}$$

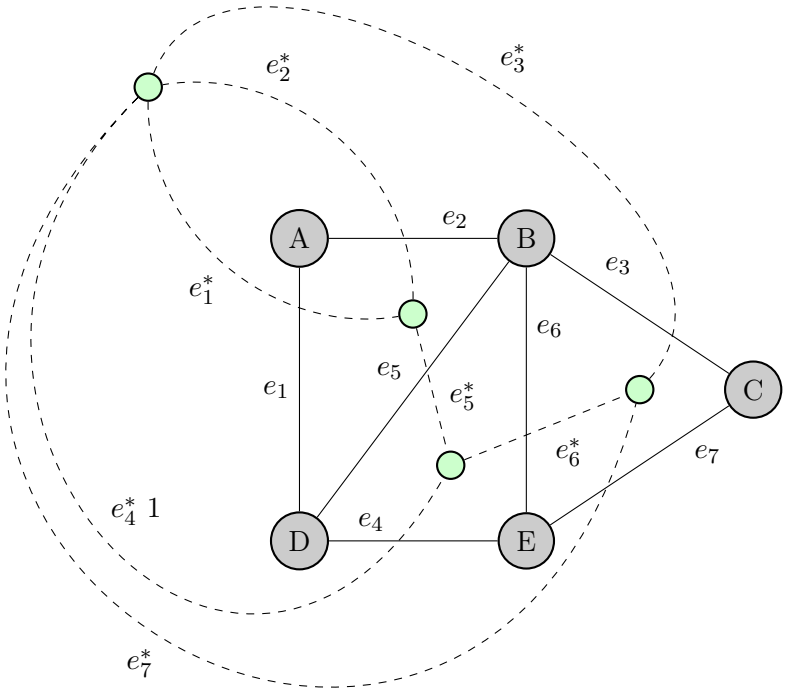
$$\implies (k + f)(mod 2) = (k_1 + f_1 + k_2 + f_2 + 1)(mod 2) = 1 \tag{4.2.72}$$

If it holds for two faces, it trivially holds for any number of faces. □

Theorem 4.41 (Kasteleyn's Theorem)**.** *Let $G \in \mathfrak{G}$. Assume G admits an orientation $\vec{\mathscr{E}}$ such that for each internal face the number of forward clockwise edges is odd. Then, $\vec{\mathscr{E}}$ is Pfaffian.*

Proof. To show that $\vec{\mathscr{E}}$ is Pfaffian, consider an even cycle c such that $G \setminus V[c]$ has a perfect matching. By the graph being planar, the number of vertices inside c is even. Indeed, even cycles enclose even cycles [17] and this requirement is strict as they must be perfectly matchable [16]. By Lemma 4.40, the number of forward edges is odd. Then, by Lemma 4.37, since the number of backward edges is odd, the orientation $\mathscr{E}$ is Pfaffian. □

Figure 4.2: G full lines, $\mathcal{G}$ dotted, credits: Zarko StackLatex



This result logically implies that computing the number of perfect matchings for a planar graph is not *#P-complete*. Indeed, it suffices to build efficiently a Pfaffian Orientation, determine its custom adjacency matrix $A^\star$ as in 4.2.54, and evaluate $det(A^\star)$. This procedure is encapsulated in the Fisher Kastelyn Temperley Algorithm.

Before introducing the precise steps, some definitions and facts are necessary.

Definition 4.42 (Faces of a planar graph $\mathscr{F}$)**.** Given a planar graph G denote the set of its faces as $\mathscr{F}$. A planar graph can indeed be seen as a mesh of $\mathscr{F}$. Faces are adjacent if they share an edge.

Definition 4.43 (Dual of a planar graph $\mathcal{G}$)**.** Given a planar graph G its dual graph $\mathcal{G} = \{\mathscr{F}, \mathscr{E}'\}$ is a graph that has a vertex for each face of G, and an edge for each adjacent face.

Example 4.44 (A graph and its dual)**.** A graph and its dual can be found in Figure 4.2.

A nice approach to the dual property that will be exploited is found in [20].

Observation 4.45 (Acyclic subgraphs and dual)**.** *An acyclic subgraph of G planar does not disconnect $\mathcal{G}$. Any acyclic subgraph is necessarily a forest (an ensemble of $\mathcal{T}$s). Thus any face is reachable by avoiding the forest, which cannot form a cycle. Then $\mathcal{G}$ has a spanning tree, and the dual property of connectedness is acyclicity.*

Theorem 4.46 (Euler's Theorem)**.** *For a planar graph G it holds that:*

$$|\mathscr{E}| = |\mathscr{V}| - |\mathscr{F}| - 2 \tag{4.2.73}$$

Proof. Let $\mathcal{T}$ be a spanning tree in G. By Definition 4.20 it is a connected acyclyc subgraph, and it has $|\mathscr{V}| - 1$ edges. Consider its complement $G \setminus \mathcal{T}$. Its dual is connected and acyclic by Observation 4.45. Thus, its dual is a spanning tree as well, and has $|\mathscr{F}| - 1$ edges. The two spanning trees together have all the edges in G. Thus:

$$|\mathscr{E}| = |\mathscr{V}| - |\mathscr{F}| - 2 \tag{4.2.74}$$

□

Thanks to Theorem 4.46, a combination of a spanning tree and a spanning tree of the dual necessarily considers all edges. By Theorem 4.41, in order to find a Pfaffian Orientation, all faces must present an odd number of forward clockwise edges. The procedure implemented to satisfy these conditions is outlined in Algorithm 6, known as the FKT Algorithm.

Algorithm 6 FKT Algorithm

Input: G planar

Output: $\#\mathscr{M}$ for G

1: Compute the planar embedding of G
2: $\mathcal{T}_1 \leftarrow$ spanning tree of G
3: **for** $e \in \mathcal{T}_1$ **do**
   orient e arbitrarily
4: **end for**
5: $\mathcal{G} \leftarrow dual(G)$
6: $\mathcal{T}_2 \leftarrow$ spanning tree of $\mathcal{G}$
7: **for** $v \in leaves(\mathcal{T}_2)$ where $v \notin root(\mathcal{T}_2$ **do** ▷ for $root(\mathcal{T}_2)$ it is satisfied by the previous
   let e be the edge in $\mathscr{F} \equiv v$ not oriented
   orient e such that $\#clockwise$ is odd
   remove e from $\mathcal{T}_2$
8: **end for**
9: Build $A^\star$ ▷ as in Equation 4.2.54
10: evaluate $det(A^\star)$
11: **return** $\sqrt{det(A^\star)} = Pf(A^\star) = \#\mathscr{M}$ ▷ by Corollary 4.36

Informally, starting from leaves of the dual spanning tree, there will be only one edge in the remaining face. Similarly, as the dual tree gets climbed bottom-up, there will always be one edge missing to orient. This leaves a trivial choice to the algorithm: if the clockwise edges are odd, orient the edge anticlockwise, else do the opposite.

The edge orientations now are a valid Pfaffian orientation, and the matrix $A^\star$ will identify the number of matches.

Since building spanning trees is of polynomial complexity, as well as reorienting edges, building a Pfaffian orientation for a planar graph with Algorithm 6 is computationally efficient. Given a Pfaffian orientation, the determinant, which is efficient to compute as well by *Gaussian elimination*, will return the desired result.

Kasteleyn's approach can be implemented to determine the phase transition of the Ising Model, as explained in [17]. For length and time constraints, it will not be shown in this document.

Bibliography

[1] H. W. Kuhn. "The Hungarian method for the assignment problem". In: *Naval Research Logistics Quarterly* 2.1 (Mar. 1955), pp. 83–97. ISSN: 00281441, 19319193. DOI: 10.1002/nav.3800020109. URL: https://onlinelibrary.wiley.com/doi/10.1002/nav.3800020109 (visited on 06/06/2022).

[2] Rainer E. Burkard and Eranda Çela. "Linear Assignment Problems and Extensions". In: *Handbook of Combinatorial Optimization*. Ed. by Ding-Zhu Du and Panos M. Pardalos. Boston, MA: Springer US, 1999, pp. 75–149. ISBN: 978-1-4419-4813-7 978-1-4757-3023-4. DOI: 10.1007/978-1-4757-3023-4_2. URL: http://link.springer.com/10.1007/978-1-4757-3023-4_2 (visited on 06/06/2022).

[3] Bernd Gartner and Michael Hoffman. *Computational Geometry Course*. URL: https://ti.inf.ethz.ch/ew/courses/CG13/lecture/Chapter%203.pdf (visited on 06/06/2022).

[4] Alan J. Hoffman and Joseph B. Kruskal. "Integral Boundary Points of Convex Polyhedra". In: *50 Years of Integer Programming 1958-2008: From the Early Years to the State-of-the-Art*. Ed. by Michael Jünger et al. Berlin, Heidelberg: Springer, 2010, pp. 49–76. ISBN: 978-3-540-68279-0. DOI: 10.1007/978-3-540-68279-0_3. URL: https://doi.org/10.1007/978-3-540-68279-0_3 (visited on 06/06/2022).

[5] Arthur F. Veinott Jr. and George B. Dantzig. "Integral Extreme Points". In: *SIAM Review* 10.3 (July 1968), pp. 371–372. ISSN: 0036-1445, 1095-7200. DOI: 10.1137/1010063. URL: http://epubs.siam.org/doi/10.1137/1010063 (visited on 06/06/2022).

[6] Alexander Schrijver. "A Course in Combinatorial Optimization". In: (), p. 221.

[7] Romeo Rizzi. "A short proof of König's matching theorem". In: *Journal of Graph Theory* 33.3 (2000). _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/%28SICI%291097-0118%28200003%2933%3A3%3C13
JGT2%3E3.0.CO%3B2-K, pp. 138–139. ISSN: 1097-0118. DOI: 10.1002/(SICI)1097-0118(200003)33:3<138::AID-JGT2>3.0.CO;2-K. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/%28SICI%291097-0118%28200003%2933%3A3%3C138%3A%3AAID-JGT2%3E3.0.CO%3B2-K (visited on 06/06/2022).

[8] Michel Goemans. *Combinatorial Optimization Lecture Notes, polyhedral combinatorics*. URL: https://math.mit.edu/~goemans/18433S13/polyhedral.pdf (visited on 06/06/2022).

[9] Robert D Borgersen. "Equivalence of seven major theorems in combinatorics". In: (), p. 18.

[10] Harold W. Kuhn. "A tale of three eras: The discovery and rediscovery of the Hungarian Method". In: *European Journal of Operational Research* 219.3 (June 2012), pp. 641–651. ISSN: 03772217. DOI: 10.1016/j.ejor.2011.11.008. URL: https://linkinghub.elsevier.com/retrieve/pii/S0377221711009957 (visited on 06/06/2022).

[11] Michel Goemans. *Combinatorial Optimization Lecture Notes, matching*. URL: https://math.mit.edu/~goemans/18433S13/matching-notes.pdf (visited on 06/06/2022).

[12] Claude Berge. "Two theorems in graph theory". In: *Proceedings of the National Academy of Sciences* 43.9 (Sept. 15, 1957). Publisher: Proceedings of the National Academy of Sciences, pp. 842–844. DOI: 10.1073/pnas.43.9.842. URL: https://www.pnas.org/doi/10.1073/pnas.43.9.842 (visited on 06/06/2022).

[13] James Munkres. "Algorithms for the Assignment and Transportation Problems". In: *Journal of the Society for Industrial and Applied Mathematics* 5.1 (1957). Publisher: Society for Industrial and Applied Mathematics, pp. 32–38. ISSN: 0368-4245. URL: https://www.jstor.org/stable/2098689 (visited on 06/06/2022).

[14] Giovanni Righini. "Complements of Operations Research". Type: presentation. Aug. 2, 2018. DOI: 10.1287/fa0f35b9-6bcb-4f07-ab71-78a7c9014fb2. URL: http://pubsonline.informs.org/do/10.1287/fa0f35b9-6bcb-4f07-ab71-78a7c9014fb2/full/ (visited on 06/06/2022).

[15] Rainer E Burkard. "Admissible transformations and assignment problems". In: (), p. 15.

[16] Michel Goemans. *Advanced Combinatorial Optimization Lecture Notes*. URL: https://math.mit.edu/~goemans/18438S14/lec4-algmat.pdf (visited on 06/12/2022).

[17] Cristopher Moore and Stephan Mertens. *The Nature of Computation*. Oxford University Press, Aug. 11, 2011. ISBN: 978-0-19-923321-2. DOI: 10.1093/acprof:oso/9780199233212.001.0001. URL: https://oxford.universitypressscholarship.com/view/10.1093/acprof:oso/9780199233212.001.0001/acprof-9780199233212 (visited on 06/07/2022).

[18] R. E. Burkard and U. Zimmermann. "Weakly admissible transformations for solving algebraic assignment and transportation problems". In: *Combinatorial Optimization*. Ed. by M. W. Padberg. Mathematical Programming Studies. Berlin, Heidelberg: Springer, 1980, pp. 1–18. ISBN: 978-3-642-00802-3. DOI: 10.1007/BFb0120884. URL: https://doi.org/10.1007/BFb0120884 (visited on 06/08/2022).

[19] L. G. Valiant. "The complexity of computing the permanent". In: *Theoretical Computer Science* 8.2 (Jan. 1, 1979), pp. 189–201. ISSN: 0304-3975. DOI: 10.1016/0304-3975(79)90044-6. URL: https://www.sciencedirect.com/science/article/pii/0304397579900446 (visited on 06/06/2022).

[20] David Eppstein. *Euler's Formula*. URL: https://www.ics.uci.edu/~eppstein/junkyard/euler/interdig.html (visited on 06/12/2022).