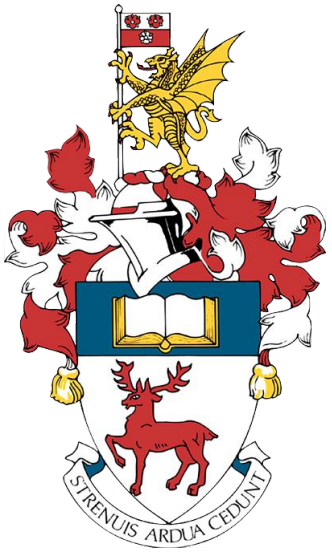
UNIVERSITY OF SOUTHAMPTON

Faculty of Social Sciences
Mathematical Sciences

Formulations and Solution Methods for a Mixed-Integer Non-Linear Bilevel Pricing Problem



by

Karl-Matthias Steinborn-Busse

A thesis for the degree of
Doctor of Philosophy

August 2022

University of Southampton

Abstract

Faculty of Social Sciences
Mathematical Sciences

Doctor of Philosophy

Formulations and Solution Methods for a Mixed-Integer Non-Linear Bilevel Pricing Problem

by Karl-Matthias Steinborn-Busse

Since the turn of the 21st century, interest in bilevel optimisation has been rapidly increasing. With a vast array of bilevel applications in energy and transport industries, interest has always existed, however, the solution methods, combined with our computational powers, meant many sizeable bilevel problems were out of reach. Most solution methods focused on very simplistic cases, continuous variables with linear constraints, whereas many problems of interest included non-linear constraints along with integer variables.

In this thesis, we study a Mixed-Integer Non-Linear Bilevel Problem, MINLBP, based upon a commodity pricing problem. The objective of the followers is to purchase a set of commodities, that satisfy some combinatorial constraints, at the minimum cost while the leader wishes to maximise their profit by introducing a taxation to commodities. We shall present three formulations; unit supply and single follower, unit supply and multiple followers and non-unit supply and multiple followers. Unit and non-unit supply relates to the quantity of each commodity available for both the leader and follower to purchase. As it shall be shown, the unit and non-unit supply scenarios must be treated differently as it drastically affects the reaction of the follower. With the single supply case, should a feasible follower reaction include commodity j, then they shall purchase that commodity. However, in the non-unit supply case, the follower can select the cheapest version of commodity j available to them. Thus, should the leader have applied a taxation to one version of j, then the follower would purchase an untaxed version instead. As a result we observe that, as the result of a max operator in both the leader and follower objective functions, we can reformulate the problem, using two methods that we call the $\bar{y}$ and γ reformulations. Following this, we shall linearise our formulations to produce a Mixed-Integer Linear Problem, MILP, using McCormick envelopes, along with binary expansions when necessary.

We then address various solution methods to be used in conjunction with the formulations described. We discuss how a simple cutting-plane approach, used within a branch-and-cut framework can easily be implemented for the unit supply case, but cannot be directly applied for the more general non-unit supply case. This is a result of the followers feasible region no longer being completely independent of the leaders variables. Therefore, we demonstrate four solution methods for the general case; two of these use a variable generation approach to introduce variables, which ensure the value function constraint is only active for the necessary sections of the feasible region, with the latter two using a sophisticated branching strategy to achieve the same results without the need for any additional variables.

We also present further solution methods focused on solving instances where the followers are restricted to a subset of responses. The first solution method is split into two parts. The first of which iteratively solves the case where the followers must respond from a set Y, adding reactions to Y with every bilevel infeasible solution. The second

stage then solves the case where we assume that the followers respond with a solution outside of Y. Following this, we shall present a branching strategy which aims to encompass this solution method within a single Branch-and-Cut framework.

Three sets of instances are generated to compare the performances of both the formulations and solution methods. Each focus their attention on certain parameters of the bilevel problem including the number of commodities, the maximum taxation and the leaders budget.

Contents

List of Figures

List of Tables

Definitions and Abbreviations

MILP	Mixed Integer Linear Problem
MINLBP	Mixed Integer Non-Linear Bilevel Problem
F^U	Leader Objective
F^L	Follower Objective
G^U	Leader Constraints
G^L	Follower Constraints
ψ	Optimal Reaction Set
ϕ	Optimal Value Function
x	Taxation applied by the leader
$\bar{x}$	Leaders binary decision to purchase commodities
y	Followers binary decision to purchase commodities
B	Leader budget constraints
M	Maximum taxation
n	Number of commodities
$\mathcal{J}$	Indices for commodities
$\mathcal{I}$	Indices for groups of followers
d	Number of followers in each group
s	Supply for each commodity
$\bar{y}$	Followers binary decision to purchase commodities from the leader
γ	Follower binary decision if they have to purchase commodities from the leader
z	Amount of each commodity the follower shall purchase from the leader
$\mathcal{MC}(x_3, x_1, x_2)$	McCormick constraints where $x_3 = x_1 x_2$
DMC	Direct McCormick Envelopes
SMC	Summation McCormick Envelopes
DI	Dichotomic
MV	Max Value
DM	Double MINLBP
$\mathcal{Y}^{Sol}$	Set of follower responses
SFCP	Symmetry Free Cutting Plane
NSFCP	Non-Symmetry Free Cutting Plane
nB	n-ary Branching
KSB	Known Solution Branching

Acknowledgements

Firstly, I would like to thank my supervisors Joerg Fliege and Stefano Coniglio. Throughout my PhD they have provided me with guidance for all aspects of PhD life, without which, I would not have been able to complete my PhD and I shall always be grateful for.

I would like to thank all of my fellow PhD students, of which there are too many to name, for the endless coffee/crossword/lunch breaks that made the Ketley room such an enjoyable place to be. I would like to especially thank Ruth, Simos, Laura, Tom, Marton and Walton (Young CORMSIS plus Tom) for talking football, replacing my 'Redding' mug and making my PhD experience one that I will always have fond memories of.

To Joe, Alex and Saif of Smallspark Space Systems. The chance to work with you in my first **proper** job not only allowed me to see how OR worked in the real world, but also gave me the motivation to finish my PhD and get it over the line so that I can get back to working with you full time.

Thanks to my parents: Mum for always being there to help me whenever I may ask for it (especially when I block a drain or dont know how to cook something) and Dad for constantly pushing me, both academically and with football. To my fiancée Jess. We started my PhD watching Jeremey Kyle on beanbags and now we are planning our wedding together. We have both come a long way over the last four years and I cannot wait to spend the rest of my life with you. These three people are the bedrock of my life and without them I honestly don't know what I would do.

To . . .

Chapter 1

Introduction

Standard mathematical programs consider problems in which there exists a single decision maker, aiming to optimise a single set of variables. Bilevel optimisation is a specific branch of optimisation, where the variables are partitioned into two sets. These variables values are decided in a sequential manner, from individual decision makers, traditionally known as the leader and follower, or upper and lower levels, each with their own constraints and objective functions. The leader determines the values of their variables first, with the follower responding to the leaders actions by solving their own problem, which is parameterised by the leaders variables. The inclusion of the follower dramatically increases the complexity for bilevel programs compared to standard mathematical programs, as points are only considered bilevel feasible if they represent an optimal response for the follower. Additionally, the introduction of discrete variables, along with non-linear terms, levitate the complexity even further. In this thesis, we aim to formulate and solve a Mixed-Integer Non-Linear Bilevel Problem, MINLBP, based upon a pricing problem. Firstly, we shall provide non-linear formulations of the problem before using a combination of binary expansions and McCormick envelopes to provide exact linear formulations, before demonstrating how sophisticated solution techniques can be used with current readily available state-of-the-art solvers. In the rest of this chapter, we give a brief background to bilevel optimisation, whilst discussing our objectives and aims of this thesis.

Bilevel optimisation, or Stackelberg games, can be dated back to 1934, when Heinrich Freiherr von Stackelberg discussed game theory in an economics setting [158]. Stackelberg games consist of multiple players, each of which act in a sequential manner, with the aim of optimising their own objectives. The case just two players, the leader and follower, was first described as a bilevel problem by J.Bracken and J.McGill in 1973 [26], who discuss bilevel optimisation with the application to a military setting. Although stemming from the 1930's, bilevel optimisation did not receive a significant amount of attention during the 20^{th} century, largely due to the computational resources available at the time. The majority of research papers focused on bilevel problems with specific

properties, such as linearity [163, 22, 33, 11, 108, 86, 114, 52, 52, 5, 20, 166, 124, 107, 66, 155, 38, 110], and any computational results had to be scaled down to small size problems. However, with the advancement of technology, the pool of solution methods and feasible applications has increased.

As already mentioned, bilevel problems are significantly harder to solve than standard mathematical programs [91, 14, 57], resulting in a large proportion of the literature, and solutions methods, focusing on cases with assumptions, such as convexity and continuity. As a result, in general, problems which include binary, or integer variables, in the lower level, have far fewer solution methods available. This is clearly an issue, given that we can find an abundance of real world instances, which can be formulated as a Bilevel Problem with discrete variables in the lower level.

In this thesis, we shall formulate a bilevel pricing problem, based around the existence of a set of commodities, which the follower wishes to purchase, to satisfy some combinatorial constraints. Meanwhile, the leaders objective is to apply some form of taxation, by purchasing these commodities and selling them to the follower at an inflated price, whilst conforming to their own budget constraints. We start with the basic formulation where there is a single follower and commodities are unique with no duplicates. This has the property that the followers feasible region is independent of the leaders variables. We then build up to the general case with multiple followers and commodities can have a supply greater than 1, which we shall call the non-unit supple case. When there are multiple versions of each commodity, should the follower wish to purchase commodity j as part of their reaction, then they have s_j many versions of j to purchase from and shall select the cheapest version. Thus, we discover that the followers feasible region is perturbed by the leaders variables and therefore, must be reflected in the solution methods used.

Clearly, one of the most relevant subjects with bilevel problems, are price setting problems [63, 98, 85, 59, 55, 67, 30, 59, 27]. . Due to the sequential nature, a majority of problems that can be described as one party defining prices for commodities, whilst another party decides which commodities to purchase can be formulated as a bilevel problem. Possibly the most common example of this, is the network pricing, or toll setting problems [85, 98, 55], where the leader can apply a taxation to a subset of arcs on a graph, whilst the followers objective is to travel between their origin and destination nodes in the cheapest way possible. In this scenario, the followers feasible region can be found to be totally unimodular and reformulation techniques, usually reserved for instances with non-discrete follower variables, can be applied to generate the single-level reformulation used with solution methods. However, in general, this is not always the case. For a more general approach, we wish to look at the case where we only know the existence of the followers constraints and nothing about their structure, nor their properties. With this model, we aim to generate linear formulations along with solution methods which can be used with state-of-the-art solvers such as CPLEX.

This thesis is structured as follows; in Chapter 2, we provide the reader with a literature review on bilevel optimisation. We begin by outlining basic definitions used to help with the understanding and formulation of our bilevel problem. Following this we discuss three applications of bilevel optimisation: Facility Location; Interdiction and Pricing problems, which appear most frequently in the literature, before outlining some fundamental bilevel properties. We then present some existing solution methods, starting with the continuous case, focusing on reformulation and enumeration techniques, followed by integer solution methods.

In Chapter 3, we provide formulations for the bilevel pricing problem's which can be described as MINLBP's. We begin with the simplest case where there is a unit supply and a single follower, increasing the complexity to a non-unit supply and multiple followers who act collectively together rather than competing against each other. In the literature, multiple followers are commonly mentioned, however a non-unit supply, causing the followers to decide between two identical commodities with varying costs is rarely discussed. Here, we discover a max operator in the leader and followers objectives, which causes us to perform further reformulations. The formulations up to this point shall be non-linear, however, we generate the linear versions by using combinations of binary expansions and McCormick envelopes. Furthermore, we discuss a cutting-plane reformulation method used in [53] and show how using McCormick envelopes could have tightened the feasible region. We then make an observation about how $\sum_{j=1}^{n}\sum_{i=1}^{m} y_{ij}x_j$ can be linearised with McCormick envelopes in two ways, which we shall describe as Summation and Direct.

In Chapter 4, we outline the various solution methods that can be used for the problem formulated in Chapter 3, distinguishing between the unit and non-unit supply cases. For the unit supply case, we demonstrate how a simple cutting-plane procedure can be used in conjunction with the "standard" branch-and-cut framework commonly used with Mixed Integer Bilevel Problems. Following this, we shall show that by having a non-unit supply, this cutting-plane approach can no longer be directly applied, as the cutting planes are no longer globally valid, because of the followers' feasible region becoming dependent on the leaders variables. As a result, this leads us towards four solution methods. The first two are variable generation approaches. As the cutting planes are only valid under specific circumstances, we introduce a series of variables which indicate when such cutting planes can be applied. In this setup, we also discuss how the existence of symmetric solutions can lead to a high number of variables being generated, even though the followers reactions are somewhat "unchanged". This brings us to introduce a symmetry-free method, where the cutting plane focus on the commodities rather than the exact responses of the followers. For the latter two solution methods, instead of introducing additional variables, we perform a tailored branching strategy which partitions the feasible region into multiple sub-regions and only one of which is the cutting plane valid and is thus applied locally.

Additionally, we also discuss three solution methods that can take advantage of the leader having pre-computed a subset of follower solutions. The first solution method, has two stages, the first of which we assume the leader knows a predefined subset Y of possible follower solutions and the follower responds with a solution from this set. This acts as a relaxed formulation of the original bilevel problem, therefore we must then solve the case where the follower responds with a solution not from Y. The second stage can be solved by using the cutting plane or branching strategies just mentioned, with the additional constraints that the followers response must not be contained in Y. However, for the first step, as the follower must respond from a pre-defined subset, we can formulate the bilevel problem such that the follower's feasible region is totally unimodular. As a result of this characteristic, we can use reformulation techniques such as the KKT or strong-duality conditions, which were previously unavailable. The final solution method, tries to encompass these two stages into a single formulation, by performing a Known-Solution-Branching, KSB, strategy, where $|Y| + 1$ many child nodes are created from the root node, with $|Y|$ many fixing the followers response to one in Y and the final node giving the follower free choice. This KSB strategy can be also be used in conjunction with the four solution methods discussed in the previous paragraph.

In Chapter 5, we present the computational tests conducted along with their results. Three sets of instances were generated to compare the performances of the formulations along with their solution methods, as well as demonstrating how the problems parameters affect their performance. In the first set, we vary the number of commodities along with the supply for each commodity. In the second set, we shall modify the number of followers and the supply of each commodity, which will allow us to compare the Direct and Summation McCormick formulations. Then, finally in the third set, we shall adjust the leaders budget, B and the maximum taxation the leader can apply M.

Finally, Chapter 6 contains our conclusions and discusses future research avenues arising from this work. Appendix A provides the reader with the linear formulations discussed throughout with [149] providing results and additional plots receptively from our computational study in Chapter 5.

Chapter 2

Literature Review

In this chapter, we shall provide the reader with a background into the literature which focuses on bilevel optimisation. We shall begin by giving basic definitions and properties commonly used when discussing a bilevel problem, which help to understand the solution methods discussed later. Following this we shall highlight three applications where bilevel optimisation appears regularly, including pricing problems, which is the basis of this thesis. Section 2.3 shall focus on existing solution methods used for solving bilevel problems with continuous variables. Although the formulations generated in Chapter 3 are for a Mixed-Integer Bilevel Pricing Problem, we a present solution method in Chapter 4 which assumes a totally unimodular lower level feasible region, which allows for the use of solution methods such as the KKT reformulation which are most commonly used for continuous bilevel problems. Lastly we move onto Integer Bilevel Problems, discussing how integer variables affect the inducible region along with their solution methods.

2.1 Properties of Bilevel Problems

Bilevel optimisation can be dated back to 1934 when Stackelberg games (non-cooperative sequential games) were introduced, with applications to economics [158]. Stackelberg games consist of multiple players, each of which act in a sequential manner, with the aim of optimising their own objectives. In the simplest case, where there are two players, the first decision maker is referred to as the leader, with the second being the follower. This first described as a bilevel problem by Bracken and McGill in 1973 [26]. The general bilevel problem, BLP, can be formulated as

$$\max_{x,y} \quad F^U(x,y) \tag{2.1a}$$
$$\text{s.t.} \quad G_i^U(x,y) \leq 0 \quad \forall i, \tag{2.1b}$$
$$y \in \arg\min_y \{F^L(x,y) : G_j^L(x,y) \leq 0 \quad \forall j\}, \tag{2.1c}$$

where F^U and G^U are the leader's objective and constraints respectively and F^L and G^L are the follower's objective and constraints respectively. Using (2.1), we can define the constraint region as

$$\Omega = \{(x,y) : G_i^U(x,y) \leq 0, G_j^L(x,y) \leq 0 \quad \forall i,j\}, \tag{2.2}$$

which is the set of points that satisfy both the leader's and the follower's constraints. Using Ω, we can define the projection onto the leader's feasible region as

$$\Omega_x = \{x : \exists y \text{ s.t. } (x,y) \in \Omega\}, \tag{2.3}$$

which is the set of possible leader variables, such that there exists a feasible response for the follower. Given that the follower's problem is parameterised by the leader's variables, for every x we define the follower's feasible region as

$$\Omega(x) = \{y : G_j^L(x,y) \leq 0 \quad \forall j\}. \tag{2.4}$$

Using (2.3) and (2.4), we can rewrite the BLP, still in a two level format, as

$$\max_{x,y}\{F^U(x,y) : x \in \Omega_x,\, y \in \arg\min_y\{F^L(x,y) :\, y \in \Omega(x)\}\}. \tag{2.5}$$

For a point (x,y) to be feasible for (2.1), we must have that y is the optimal solution for the follower's problem. Therefore, for a given x, we can define the set of reactions that are optimal for the follower as

$$\psi(x) = \{y : y \in \arg\min_y\{F^L(x,y) : y \in \Omega(x)\}\}, \tag{2.6}$$

and the optimal value function as

$$\phi(x) = \min_y\{F^L(x,y) : y \in \Omega(x)\}. \tag{2.7}$$

Although Ω contains the points that are feasible for the constraints, this is not equivalent to the feasible region of (2.1). This is given by

$$\text{IR} = \{(x, y) : x \in \Omega_x,\, y \in \psi(x)\}, \tag{2.8}$$

which, is labelled as the inducible region [13], the set of all points (x, y) such that for the leaders variable x, there exists some follower response and y is the optimal response of the follower. Alternatively, we could have defined the inducible region as $\text{IR} = \{(x, y) : x \in \Omega_x, y \in \Omega(x),\, F^L(x, y) - \phi(x) = 0\}$, where we focus on the followers objective values rather than their reaction set. For any follower reaction $y \in \psi(x)$, we must have $F^L(x, y) = \phi(x)$, so clearly these two descriptions of IR are the same. (2.1) can therefore be rewritten as the single level problem

$$\begin{aligned} \max_{x,y} \quad & F^U(x, y) && \text{(2.9a)} \\ \text{s.t.} \quad & (x, y) \in \text{IR} && \text{(2.9b)} \end{aligned}$$

Figure 2.1 highlights how the inducible region and Ω can differ [14]. Here, the problem being solved is given by (2.10). The constraint region Ω is outlined by both the solid and the dashed line, whereas the inducible region IR, is the piecewise linear dashed line. We can see that the inclusion of the followers objective function has dramatically altered the optimal solution. Without the follower's objective the optimal solution can be found at $(2, 4)$ with an objective value of -42. Whereas, with the follower's objective the optimal solution is at $(8, 1)$ with an optimal value of -18.

$$\begin{aligned} \min_{x} \quad & -x - 10y && \text{(2.10a)} \\ \text{s.t.} \quad & y \in \arg\min_{y}\{y : && \text{(2.10b)} \\ & \text{s.t.} \quad -25x + 20y \leq 30 && \text{(2.10c)} \\ & \qquad x + 2y \leq 10 && \text{(2.10d)} \\ & \qquad 2x - y \leq 15 && \text{(2.10e)} \\ & \qquad 2x + 10y \geq 15\}. && \text{(2.10f)} \end{aligned}$$

Even when all functions are linear and the variables are continuous, the BLP is extremely difficult to solve even for relatively small problem sizes. When the leader and follower problems admit convexity, the resulting BLP can still be non-convex and was

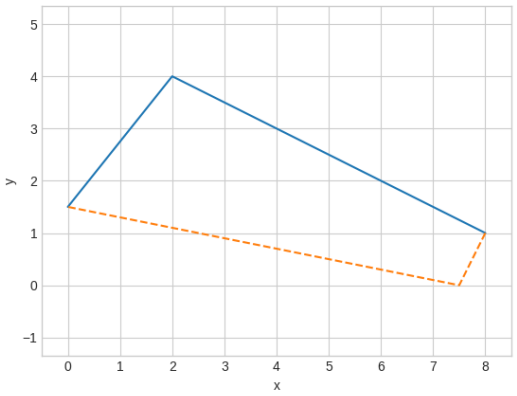


FIGURE 2.1: Example of how the inclusion of the followers objective alters the optimal solution [14].

shown to be NP-Hard by [91] and [14] with the mixed integer bilevel linear program shown to be $\sum_2^P$-hard [57].

As can be seen in [145], interest in bilevel optimisation has dramatically increased since the start of the 21st century. As a result, solution methods have developed over time, with reviews of solution methods [157, 42, 145, 95, 163, 49] showing the advancements being made.

These reviews provide a good insight into how not only methods are being developed for solving BLP's, but also the instances of problems that can be attempted given the improvement of methods. [95] is one of the earliest reviews of bilevel optimisation, covering both applications and solution methods. They acknowledge that most of the algorithms present at the time can be partitioned into three categories; extreme point search, Karush-Kuhn-Tucker (KKT) and descent methods. They note that without 'significant restrictions' on the follower's problem, due to the lack of good solution methods, it will be difficult to obtain a global optimum, possibly one of the reasons why problems containing integer variables have not been discussed here.

The review [163] also disucss Vertex Enumeration, along with the KKT approach, whilst briefly mentioning work being done on discrete bielvel programs. [112, 165] both provide branch and bound frameworks, specifically for BLP that contain binary variables. Additionally, [165] produces a heuristic, which can solve the binary BLP to near optimality in linear computation time as the number of binary leader variables increases linearly.

[145] introduce how evolutionary methods can be used for solving BLP, focusing on three specific branches: nested, single level reduction and metamodeling.

[66] originally developed theorems and properties for the maxmin problem

$$\max_x \min_y \{F(x,y) : G(x,y) \leq 0 \quad \forall j\}. \tag{2.11}$$

More specifically, they focus on the linear maxmin problem, where $F(x,y) = cx + dy$ and the constraints are $By \leq b - Ax$. [13] extended these to the linear bilevel problem, LBLP, given by

$$\max_{x \geq 0} \quad a^\top x + b^\top y \tag{2.12a}$$

$$\text{s.t.} \quad \max_{y \geq 0} \{c^\top x + d^\top y : \; Ax + By \geq \bar{b}\} \tag{2.12b}$$

The following theorems about the geometry of the solution space for (2.12) can be found in both [10] and [13]

Theorem 2.1. *The LBLP is equivalent to maximising $ax + by$ over a feasible region comprised of a piecewise linear equality constraint.*

Given that the follower reacts after the leader, we can assume $c = 0$. Therefore, the followers problem becomes $\max_y \{dy : By \geq \bar{b} - Ax, y \geq 0\}$. As we can see, this is a linear program and therefore, should we not have unboundedness and the constraints admit a feasible solution, then the solution occurs at a vertex of a piecewise linear function. Thus, the solution to the LBLP must satisfy $\max_y \{dy : By \geq \bar{b} - Ax, y \geq 0\} - dy = 0$, which is a piecewise linear equality constraint. This can also be seen in Figure 2.1 where the optimal solution lies on the dashed line which is given by $2x - y - 15 = 0$ for $x \in [0, 7.5]$ and $2x + 10y - 15 = 0$ for $x \in [7.5, 8]$.

Corollary 2.2. *A solution to the LBLP occurs at a vertex of IR.*

This result was given in [21] who also stated that the LBLP's feasible region could be described as $(x,y) \in$ coIR, where coIR is the convex hull of the inducible region.

Theorem 2.3. *The solution of the LBLP occurs at a vertex of Ω.*

The proof to this is stated in both [21] and [10]. Following this, they authors note that any vertex of the inducible region is also a vertex of Ω, leading to Corollary 2.4. From this result, algorithms have been developed to solve LBLP using extreme point search procedures.

Corollary 2.4. *If x is an extreme point of IR then it is an extreme point of Ω.*

Throughout [13], they have assumed the existance and uniqueness of a solution, however this is not always the case. As such, if $|\psi(x)| > 1$ there may be a follower solution

which the leader would prefer the follower to choose, as it will give the leader a better objective. This would imply that there is some cooperation between the leader and follower, possibly through some side payments [167], which is somewhat contradictory to the structure of bilevel optimisation. These scenarios, where the leader can determine which response from $\psi(x)$ the follower takes, are defined as Optimistic, or strong, bilevel optimisation.

In contrast, for Pessimistic, or weak, bilevel optimisation, the leader anticipates the worst case scenario, where the follower selects the response in $\psi(x)$ that gives the leader the worst objective. The BLP is then rewritten as a three level problem

$$\max_x \min_y \{F^U(x,y) : x \in \Omega_x, y \in \Omega(x), y \in \psi(x)\}. \tag{2.13}$$

Modelling a problem as either an Optimistic or Pessimistic one can drastically affect the optimal solution. [167] demonstrates the contrast in objective functions values with the example $\min_x\{x : x \geq y, y \in \arg\min_y\{-y^2 : y \in [-1,1]\}\}$. As the followers problem is independent of the leaders variables, the leader is forced to give x a value of at least 1, resulting in an optimal objective value of 1, in the Pessimistic case. Whereas, in the Optimistic setting, x must have a value of at least -1, giving an optimal objective value of -1. [154] focus their attention to the independent Pessimistic problem, where the feasible region of the leader and follower are separate from each other.

It is assumed that the leader has full knowledge of the followers problem, however we do not assume the reverse. Should the leader contain constraints that include both the leader and follower variables, known as coupling constraints, then there is a possibility that the followers optimal response to x is one that violates some G_i^U. Therefore, it is the responsibility of the leader to ensure that the followers optimal reaction is upper-level feasible. One may assume that any coupling constraints can be shifted from the upper level to the lower, to ensure feasibility. However, Figures 2.2 and 2.3 show how the feasible points before and after shifting constraints are not necessarily the same and that the optimal solution prior to the shifting can potentially be no longer feasible. Here, the bilevel problem is given by (2.14), which can be found in [110], and has two coupling constraints, (2.14b) and (2.14c), in the leaders feasible region. When these constraints are part of the leaders problem, the inducible region is disconnected, with the optimal solution found at C, $(8,6)$, giving a leader objective value of -20. Should the leader have chosen a value of x such that $3 < x < 8$, the follower would have reacted with a y, which would be above the triangle ABC. This would have been an infeasible solution for the leader, as at least one of their constraints would be violated, which means they have to select an x such that $x \leq 3$ or $x \geq 8$. However, when (2.14b) and (2.14c) are moved to the follower's problem, the inducible region is no longer disconnected, the leader can select and value of x and the optimal solution can be found at B, $(6,8)$, with a leader objective value of -22.

$$\min_{x} \quad -x-2y \tag{2.14a}$$
$$\text{s.t.} \quad 2x-3y \geq -12 \tag{2.14b}$$
$$x+y \leq 14 \tag{2.14c}$$
$$y \in \arg\min_{y}\{-y : \tag{2.14d}$$
$$-3x+y \leq -3 \tag{2.14e}$$
$$3x+y \leq 30\}. \tag{2.14f}$$

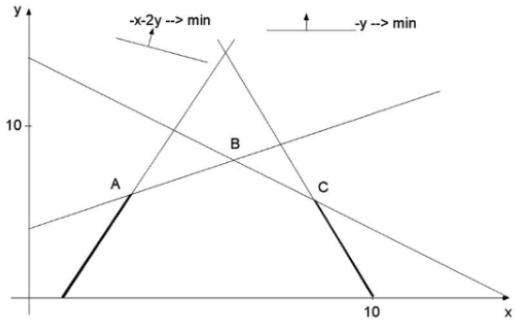


FIGURE 2.2: Feasible region when the linking constraints are in the leaders problem.

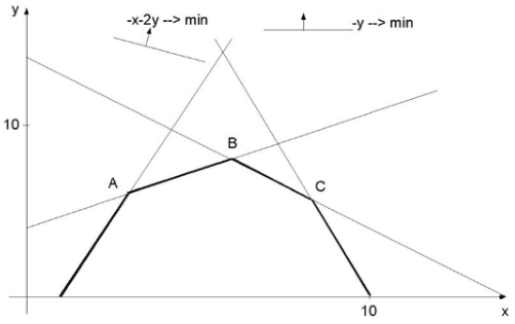


FIGURE 2.3: Feasible region when the linking constraints are in the followers problem.

2.2 Applications

As a result of the sequential nature of multi-level programs, we can find an abundance of applications which can be modelled using the bilevel framework. [50] gives a comprehensive list of applications that have been investigated thus far. Examining this list, we can see that Electricity Markets and Networks, Facility Location and Production Problems, Problems over a Network and Interdiction Problems have received significant attention.

2.2.1 Facility Location

As discussed in [47], one of the earliest works in facility location can be found in [162], who consider the introduction of a single facility to minimise the distance between facility and customer. Applications of facility location problems include emergency medical service bases [131, 92], warehouse location [171, 80], school bus routing [123] and waste management [135].

There are two types of facility location problems, classical and competitive. In classical facility location problems, the leader is planning on introducing a number of facilities to the market, with the objective of minimising their costs, which combine the prices of creating the new facilities plus some costs relating to the customers for each facility. The customers, who act as the followers, will have a preference as to which facility they will use [34, 82, 83], a parameter usually based upon, but not limited to, distance. [39, 40] provide theorems related to valid inequalities and facets within the location problem, with [75] presenting a branch and peg algorithm and [111] heuristically solving the dual of the relaxation.

The competitive facility locations problems are the same as classical, with facilities being introduced and the follower selecting their preferred option. However, we now allow for competitors in the decision space of the follower, meaning the leader's facilities will have to compete for the market [125]. Within this subsection of facility location problems, there exist further subsections: static competition, competition with foresight and dynamic competition [127].

2.2.2 Interdiction Problems

Problems where the leader and follower have the same objective function, but wish to optimise in opposite directions, are often labelled as interdiction, or maxmin problems. Examples of these problems include the shortest path interdiction problem [57, 90], the binary knapsack interdiction problem [179] and project interdiction problems [29]. From this list of examples, we find applications to networks [148, 153], where the leader intends to hinder the followers objective by the removal of arcs and/or nodes [168].

Applications of interdiction problems can be found in military settings, where the leader aims to hinder enemy movement and effectivness by means of destruction or damage. [169] gives examples of interdictions problems relating to a military setting, dating as far back as 479BC.

2.2.3 Pricing Problems

As a result of their sequential nature, pricing problems cover a large area of the applications of bilevel optimisation. In general, the leader wishes to determine the price of some commodities with the aim of maximising their revenue, knowing that the follower will purchase items that minimise their objective, usually total cost. An early example of a pricing problem can be found in [22], who focus on a linear pricing problem. A classical example of this type of problem is the toll setting problem, [63, 98, 100, 99, 85, 59], where the leader determines some tolls across a set of arcs in a network,

with the followers problem being that they wish to travel from their origin to their destination at the minimum cost. [98] show that for the case where the leader has control of a single arc, the bilevel problem can be solved in polynomial time.

Within pricing problems, there are a multitude of different leader and follower characteristics that can influence what commodities are available to the follower, along with their purchasing strategy, [152]. [60, 61, 93, 96, 139] focus their attention on models based on reservation prices, where the followers give an initial valuation of how much they are willing to pay for each product, purchasing the commodities who give them the most value for money. Envy-free pricing is where the leader also takes into account the fairness of the pricing to each customer, as studied in [27, 79, 67], however can only applicable when the supply is limited, [30].

2.3 Solution Methods

The majority of solution methods for bilevel optimisation can be partitioned into two groups, see [134]. The first of which are reformulation methods. The inclusion of the follower problem within the leaders constraints is what causes BLP to be significantly hard to solve. As a result, a large amount of research has been focused on creating a single-level problem, by giving the leader control of the follower's variables and replacing the followers problem with a set of constraints that inform the leader how the follower would react. These reformulation approaches ensure that, even though the leader has control of the follower's variables, the constraints that have replaced the follower's problem are only satisfied if the followers variables represent a true reaction of the follower. To do so, there are predominantly three approaches: Optimality Conditions; Optimal Reaction Set Mapping and Optimal Value Function [141].

By using these reformulations, the leader is left with a single-level problem, which can be solved using state-of-the-art solvers, whose solution methods have been adjusted which can be done using callbacks. However, there are drawbacks to using these methods. The mapping and function required for the Optimal Reaction Set Mapping and Optimal Value Function approaches respectively, are seldomly available. Therefore, in most cases, approximations need to be made, which are iteratively improved as a result of the previous iteration providing an incorrect reaction of the follower. Optimality conditions also have their drawbacks. There exists optimality conditions that can only be applied to follower's problems that admit a specific structure, such as linearity, meaning not all optimality conditions can be applied to all problems, unlike the Optimal Reaction Set and Value Function methods.

The second group of solution methods can be described as enumeration techniques. Such techniques can be defined as repeatidly solving programs and/or subprograms until they terminate at a solution. A large proportion of these techniques rely on the

property that the solution of an LBLP lies at a vertex of the feasible region created by taking the union of the constraints from the leaders and followers problems. From this property, numerous algorithms [12, 15, 22, 38, 122, 155] have been developed with the "Kth-Best" algorithm being the most recognisable.

2.3.1 Reformulation Methods

With the difficulty of BLP's stemming from the presence of an optimisation problem within the constraints, a large amount of research has been carried out in creating equivalent single-level problems. The goal is to generate an equivalent single-level problem, which can then be solved with existing solution methods.

Fundamentally, such reformulation methods belong to one of three categories: Optimality Conditions, Optimal Value Function and Reaction Set Mapping. With the Optimality Conditions, the aim is to replace the lower-level problem with a set of constraints that are necessary and sufficient for follower optimality. The Karush-Kuhn-Tucker, KKT, are widely used for this purpose. However, as we shall show, they are only necessary and sufficient when the followers problem meets specific requirements. Although the Optimal Value Function and Reaction Set Mapping reformulation methods technically come under Optimality Conditions, their wide spread use allows them to have their own category.

2.3.1.1 Optimality Conditions

Arguably the most common optimality conditions are the KKT conditions. These replace the follower's problem with a set of constraints that are only satisfied by a local minimum of the followers problem [56]. The single-level formulation is given by

$$\begin{aligned}
\max_{x,y,\lambda} \quad & F^U(x,y) && \text{(2.15a)}\\
\text{s.t.} \quad & G_i^U(x,y) \leq 0 \quad \forall i \in 1,\dots,q^U, && \text{(2.15b)}\\
& G_j^L(x,y) \leq 0 \quad \forall j \in 1,\dots,q^L, && \text{(2.15c)}\\
& \nabla_y L(x,y,\lambda) = 0, && \text{(2.15d)}\\
& \lambda_j G_j^L(x,y) = 0 \quad \forall j, && \text{(2.15e)}\\
& \lambda_j \geq 0 \quad \forall j, && \text{(2.15f)}
\end{aligned}$$

where

$$L(x,y,\lambda) = F^L(x,y) + \sum_{j=1}^{m} \lambda_j G_j^L(x,y), \tag{2.16}$$

and is commonly referred to as the Lagrangian. However, these KKT optimality conditions can only be applied to problems that satisfy specific qualifications such as the Mangasarian-Fromowitz constraint qualification (MFCQ) or the Linear Independence Constraint Qualification (LICQ) [94]. Along with these, [18, 19] showed how we can also include Cottle, Abadie, Kuhn-Tucker, Zangwill, Arrow-Hurwicz-Uzawa and Slatter constraint qualifications to this list.

From [177] and [51] we get the following theorem.

Theorem 2.5. *Let G^U be independent from y and $F^L(x,\cdot)$ $G_j^L(x,\cdot)$ for all j be convex and $\mathcal{C}^1$ for all $x \in \Omega(X)$. Then, the following statements hold:*

(i) Let $(\bar{x},\bar{y})$ be globally (resp. locally) optimal for (2.1) and the LMFCQ be satisfied at $(\bar{x},y), y \in \arg\min\{F^L(\bar{x},y) : G_j^L(\bar{x},y) \leq 0, \forall j\}$. Then, for each $z \in \Lambda(\bar{x},\bar{y})$, the point $(\bar{x},\bar{y},z)$ is a global (resp. local) optimal solution of (2.15).

(ii) Let the LMFCQ hold at all $(x,y), y \in \arg\min\{F^L(\bar{x},y) : G_j^L(x,y) \leq 0, \forall j\}, x \in \Omega(x)$ (resp. at $(\bar{x},y), y \in \arg\min\{F^L(\bar{x},y) : G_j^L(\bar{x},y) \leq 0, \forall j\}$) and $(\bar{x},\bar{y},z)$ be a global (resp. local) optimal solution (resp. for all $z \in \Lambda(\bar{x},\bar{y})$) of (2.15), then the point $(\bar{x},\bar{y})$ is a global (resp. local) optimal solution of problem (2.1).

Here $\mathcal{C}^1$ is the class of functions whose derivates are continuous and LMFCQ refers to the lower-level Mangasarian-Fromowitz constraint qualification which holds at $(\bar{x},\bar{y})$ if there exists d such that

$$\nabla_y G_j^L(\bar{x},\bar{y})^\top < 0 \quad \forall j \in I^2(\bar{x},\bar{y}), \tag{2.17}$$

where $I^2(\bar{x},\bar{y}) = \{j : G_j^L(\bar{x},\bar{y}) = 0\}$ and

$$\Lambda(x,y) := \{z \in \mathbb{R}^{q^L} | \nabla_y L(x,y,z) = 0, z \geq 0, G^L(x,y) \leq 0, z^\top G^L(x,y) = 0\}. \tag{2.18}$$

From this theorem we learn that should the LMFCQ hold at the appropriate points, for every globally (locally) optimal solution to the original bilevel problem then the corresponding point is an optimal solution to (2.15) and vice versa. As discussed in [177], this theorem can be very sensitive to the convexity and constraint qualifications, with example 2.1 in [177] highlighting this with (2.15) not having a solution.

Likewise, should the follower's problem not satisfy these qualifications, then the solutions of the KKT conditions may not coincide with the local minima of the follower's

problem, as seen in Figure 2.4. Here, the bilevel problem is given by (2.19), and can be found in [**?**], whose optimal solution is given by $(0.957, 1)$. Taking the first-order derivative of the follower's objective function we get the equation $x = \frac{1-y}{1+y}e^{4y}$, which is plotted in Figure 2.4. In this figure, the red sections of the line represent the inducible region. However, using the KKT approach, we would get three (y, x) solutions, $(-0.98, 1.98)$, $(0.42, 2.19)$ and $(0.895, 1.99)$, with only $(-0.98, 1.98)$ belonging to the inducible region. Therefore, using the KKT approach would result in $(0.895, 1.99)$ being returned as the solution, which is neither feasible nor optimal.

$$\min_{x} \quad (x-2)^2 + (y-1)^2 \tag{2.19a}$$

$$\text{s.t.} \quad y \in \arg\min_{y}\{-xe^{-(y+1)^2} - e^{-(y-1)^2}\}. \tag{2.19b}$$

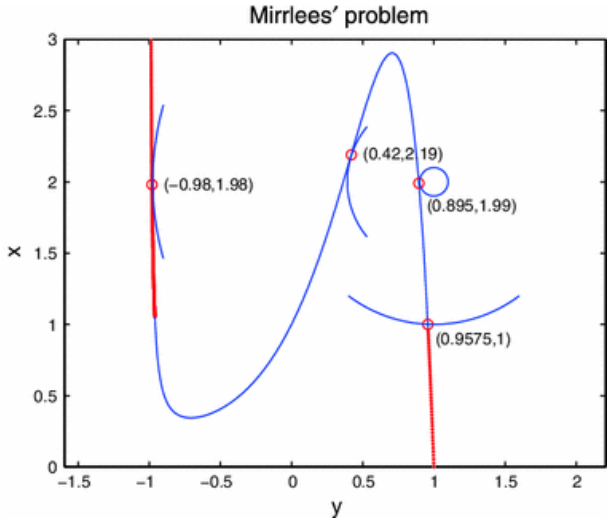


FIGURE 2.4: Example of how first-order fails with non-convex followers objectives [**?**].

(2.15) is a mathematical program with complementarity constraints, MPCC, due to $\lambda_j G_j^L(x, y) = 0$, meaning that this single-level problem is nonconvex and therefore still one which is difficult to solve [146]. Even so, there are existing methods that aim to deal with these complementarity constraints. [15, 64] use a KKT reformulation, however by dropping the complementarity constraints and solving the relaxed problem, nodes are created when the complementarity conditions are not satisfied, with one node enforcing $\lambda_j = 0$ and the other $G_j^L(x, y) = 0$, for some j. Alternatively, [73] show that by introducing additional binary variables and using big-M notation, the resulting MILP can be directly implemented using readily available state of the art solvers, such as CPLEX. However, as is the drawback with big-M's, finding the goldilocks value for M, where it is neither too large nor too small, can be a challenge in itself. [140] showed

that using Special Ordered Sets, SOS, of type 1 give the same result as the big-M formulation, without needing the corresponding big-M reformulation. SOS constraints can also be handled by state-of-the-art solvers, also making this method advantageous by its easy implementation, however the use of SOS constraints can be computationally expensive [124].

The MPCC can be treated as a nonlinear program by relaxing the complementary constraints, such that the left hand side can take any value between 0 and t for some small positive t [130, 136]. Obviously, when $t = 0$, the non-linear program is equivalent to the KKT reformulation. The parameterised non-linear program typically satisfies some constraint qualification and is therefore easier to solve [124]. Algorithms, generally, iteratively solve the nonlinear program to optimality, while converging t towards 0.

[88, 107] provide penalty function approaches. Dropping the complementary constraints and incorporating them into the objective of the leader via a penalty function, any solution that is bilevel infeasible will now appear as one with an unattractive objective function.

The algorithm presented by [124] aims to tune the big-M values by using the solutions to the relaxed nonlinear programs, similar to [130, 136]. The resulting algorithm is one which utilises the advantages described above which deal with the complementary constraints, whilst being easily implemented with readily available state-of-the-art-solvers.

An alternative approach to the KKT reformulation is to use the strong-duality condition. For every linear programming problem, there exists its corresponding dual problem. Using the relevant theorems from [10], we can see that an optimal solution of a linear program occurs when the primal and dual objectives are equal for primal and dual feasible variables. Therefore, should the followers problem be linear, it can be replaced by the strong-duality condition. [119, 8] present reformulations of bilevel problems, where the follower's problem admits convexity. Generally, using the Lagrangian dual leads to a single-level problem that is not generally differentiable due to the complementary slackness constraints. However, the authors of [119] present a reformulation using an ϵ approximation of the bilevel problem using a regularised constrained Lagrangian dual function, which has the advantage of being differentiable and satisfying the Mangasarian- Fromovitz Constraint Qualification, MFCQ.

[181] transform a weak linear bilevel program, using strong duality and show that the solution of the resulting single level, with bilinear terms, is a solution to either one of two disjoint bilinear programs. [5, 20, 166] demonstrate the use of a penalty function approach, where the duality gap is penalised in the leaders objective.

2.3.1.2 Optimal Value Function

For every leader variable x, we define $\phi(x) = \inf_y\{F^L(x,y) : G_j^L(x,y) \leq 0 \, \forall j\}$ as the optimal value function. Shown in [120], the BLP can be reformulated as

$$\max_{x,y} \quad F^U(x,y) \tag{2.20a}$$
$$\text{s.t.} \quad G_i^U(x,y) \leq 0 \quad \forall i, \tag{2.20b}$$
$$G_j^L(x,y) \leq 0 \quad \forall j, \tag{2.20c}$$
$$F^L(x,y) \leq \phi(x), \tag{2.20d}$$

where it can easily be shown that the optimal solution to (2.20) is also optimal for (2.1). Although (2.20) is a single-level problem and would appear to be easier to solve, this is not always the case. ϕ is not easily computable and in general non-differentiable. Similarly, at every feasible point, (2.20d) is an equality and therefore the constraint qualifications, such as the MFCQ, are violated [173]. As a result, [173] discuss calmness and partially calmness conditions, along with their relationship with the solution of (2.20), with partial calmness first introduced in [41]. They also demonstrate the relationship between solutions that are partially calm and penalty programs where (2.20d) is absorbed by the objective function.

[53, 97] use the value function reformulation for a specific problem type, where the feasible regions of the leader and follower depend only on their respective variables, i.e. $\Omega = \{(x,y) : G_i^U(x) \leq 0, G_j^L(y) \leq 0 \, \forall i,j\}$. As the feasible regions are independent, the solutions for the follower are never made infeasible by the leader variables. Therefore, let $Y = \{y \in \{0,1\}^m : G_j^L(x,y) \leq 0\}$, where m is the number of follower variables, be the set of all points contained in the followers feasible region, then (2.20d) can be replaced by

$$\max_{x,y} \quad F^U(x,y) \tag{2.21a}$$
$$\text{s.t.} \quad G_i^U(x,y) \leq 0 \quad \forall i \tag{2.21b}$$
$$G_j^L(x,y) \leq 0 \quad \forall j \tag{2.21c}$$
$$F^L(x,y) \leq F^L(x,y_k) \quad \forall y_k \in Y. \tag{2.21d}$$

[53, 97]'s definition of Y means that there is a finite number of follower points. Should $|Y| = \inf$, constraint (2.21d) can not be implemented and using a subset of Y would be a relaxation of the original bilevel problem. Using this formulation [97, 53, 55, 52]

present algorithms that repeatedly solve (2.21) to get the solution $(\bar{x}, \bar{y})$ and then solve the followers problem to get $\psi(\bar{x})$. If $\bar{y} \in \psi(\bar{x})$, then $(\bar{x}, \bar{y})$ is optimal, else Y is updated to include the solution $\bar{y}$ and the process is repeated. [55, 52] show that from their algorithm, not only will the accumulated point be globally optimal, but if the polyhedron from the follower's feasible region is compact, then only its vertices need to be considered and thus the algorithm shall stop after a finite number of iterations.

This method can be expanded to cases where the feasible regions of the leader or follower contain variables from the other level, however, additional checks have to be implemented as $F^L(x, y) \leq F^L(x, y_k)$ is not a valid cutting plane globally for every follower solution in Y.

For BLP's, where a subset of the lower-level variables must take integer values, many of the optimality conditions, such as KKT and strong duality, cannot be applied. Hence, the optimal value function approach is a popular reformulation as it is satisfied for both discrete and continuous problems. In [68, 70, 71, 69], Fischetti et al. present methods using the high point relaxation, HPR, where constraint (2.20d) is dropped. The work by Fischetti et al. is initially based on the work by [159, 172], who presented their Watermelon Algorithm. As a result of being able to apply the optimal value function with integer variable problems, we use this method in the solution methods found in Chapter 4

2.3.1.3 Reaction Set Mapping

For the reaction set mapping approach, rather than achieving bilevel feasibility through the followers objective value, we use rational responses of the lower level. Similar to how we defined $\phi(x)$, the optimal reaction map $\psi(x)$ is the set of reactions that are optimal for the follower, i.e. $\psi(x) = \arg\min_y \{F^L(x, y) : G_j^L(x, y) \leq 0 \, \forall j\}$. Hence the single-level formulation is

$$\max_{x,y} \quad F^U(x, y) \tag{2.22a}$$

$$\text{s.t.} \quad G_i^U(x, y) \quad \forall i, \tag{2.22b}$$

$$y \in \psi(x). \tag{2.22c}$$

Should the leader know completely how the follower will react, then this single-level problem is exact and provide the optimal solution to the original BLP. However, full knowledge of the follower's reaction is very rare [145]. [144, 142, 143] present evolutionary algorithms that approximate the lower-level optimal reaction as a function of the leaders variables. Figures 1 and 2 in [141] show how $\psi(x)$ can be approximated as both a single valued map and as a set.

Comparing the optimal value function and reaction set mapping, we find that the optimal value approach is less complicated as $\phi(x)$ shall always be a scalar value irrespective to the structure of the followers problem. Whereas ϕ can become complicated when the follower's reaction set is not singular. However, conversely, by computing ψ we no longer need to compute the followers variables as they have already been generated within ψ, something which still needs to be done with the optimal value function.

2.3.2 Enumeration Techniques

2.3.2.1 Vertex Enumeration

Discussed by [22] and earlier by [33, 11], an important property about the location of an optimal solution is presented. Let the leader and the follower objectives both be linear and let S be the feasible region defined by the constraints from the upper and lower levels. Then the optimal solutions to the linear bilevel problems occurs at an extreme points of S. [22] present four algorithms based on vertex enumeration, with the "K^{th}-Best" algorithm possibly being the most famous [33]. [83] outline a branching algorithm for a linear bilevel program, based on which constraints must be active at an optimal solution. [66] also present a vertex enumeration algorithm within a branch and bound framework.

2.3.2.2 Evolutionary Techniques

Many of the reformulation techniques that have been described thus far rely on specific assumptions, such as linearity and convexity, which limits the applications these algorithms can be applied to. However, genetic algorithms don't heavily depend on such assumptions and therefore are an alternative solution method. For example, the algorithm in [161] is able to handle a non-differentiable leader objective function and a non-convex follower's problem. One of the first evolutionary algorithms for bilevel optimisation can be found in [108], who use a genetic algorithm for the upper level and linear programming in the lower level. A similar approach can be found in [174], who instead use a Franke-Wolfe algorithm to solve the lower-level problem. Such methods are often referred to as nested methods, where the lower-level problem is solved for a population of upper-level points.

Along with nested methods, evolutionary methods can be used in conjunction with classical techniques. [86, 114] both present genetic algorithms that solve the single-level reformulation, after replacing the lower level with its KKT optimality constraints. The latter solves the program by introducing binary variables to tackle the complementary constraints. [160] similarly solve the KKT reformulation of a non-linear bilevel problem by solving a specific two-objective program using a genetic algorithm where

F^U and G^U are both non-differentiable and non-convex, whereas F^L and G^L are differentiable and convex for fixed leader variables. The multi-objective reformulation is created by the lower level being replaced with its KKT conditions, with the second objective coming as a penalty function. The standard evolutionary techniques, mutation, crossover and selection, are iteratively applied to a population. The authors note that after these processes, a population may contain points that violate one of the constraints and are therefore infeasible. To handle this, they invoke a constraint-handling procedure, which first makes the infeasible point feasible for the linear constraints, then the equality constraints and finally the non-linear constraints. [31] use a genetic algorithm along with the knowledge that a solution occurs at a vertex of the inducible region to create a genetic algorithm that solves the bilevel program with linear objective functions and IR is a polyhedron.

[102] focus on non-linear bilevel problems where F^U and F^L are non-convex, F^L is a function of the linear expressions of all variables and the constraints in G^L are convex with respect to the followers variables. They recognise that a point (x, y) is only bilevel feasible if y is the optimal response for the follower, therefore, when creating a population, there is a high chance that a large proportion of points will be infeasible. Thus, they create their own population for the leader's variables and then solve the follower's problem using a decomposition scheme, to obtain a population that is approximately bilevel feasible. From this population, they use the standard crossover and mutation techniques to generate the new population, where the fitness function continues the work by [137]

Should ψ or ϕ be known, then the bilevel program can be reduced to their corresponding single level. [142] presents an evolutionary algorithm that creates a mapping for ψ which is improved at each iteration.

The disadvantage with evolutionary techniques is the computational expense. By generating a large pool of upper-level solutions, each of which has a nested optimisation problem, the computation required for bilevel problems even with a relatively small number of variables can still be large [142].

2.4 Integer Bilevel Problems

In many of the applications that have been listed earlier, there is a need for a subset of variables to take integer values. For example, in the facility location and toll setting problems, we can find binary variables which act as decisions, whether a facility is constructed or if an arc is traversed by the follower, respectively. Similar to single-level optimisation, the inclusion of discrete variables can mean that the feasible region is no longer continuous and in general makes the problem much more difficult to solve. For bilevel optimisation, we can categorise any problem into one of four classifications:

1. Continuous-Continuous. All leader and follower variables are continuous and have no integrality constraints.
2. Continuous-Integer. All of the leader's variables are continuous, with a subset of the follower's variables having integrality constraints.
3. Integer-Continuous. A subset of leader variables have integrality constraints and all of the follower's variables are continuous.
4. Integer-Integer. A subset of both leader and follower variables have integrality constraints.

Of course, we can have Mixed-Integer Problems in both the upper and lower level, however, these classifications are just used to demonstrate how integrality constraints can dramatically alter the inducible region. Figure 2.5, [156], illustrates the inducible region for a problem where the lower level is given by

$$\begin{aligned}
\min_{y} \quad & y && \text{(2.23a)}\\
\text{s.t.} \quad & x + y \leq 2 && \text{(2.23b)}\\
& -x + y \leq 2 && \text{(2.23c)}\\
& 5x - 4y \leq 10 && \text{(2.23d)}\\
& -5x - 4y \leq 10. && \text{(2.23e)}
\end{aligned}$$

Let IR^i be the inducible region of classification i. [156] shows that $\text{IR}^3 \subset \text{IR}^1$ and $\text{IR}^4 \subset \text{IR}^2$. These two relationships are very useful for computing upper and lower bounds, similar to how bounds are discovered for standard single-level MIPs. The general mixed-integer linear bilevel program, MILBLP, can be formulated as

$$\begin{aligned}
\max_{x^C, x^I} \quad & c_C^U x^C + c_I^U x^I + d_C^U y^C + d_I^U y^I && \text{(2.24a)}\\
\text{s.t.} \quad & A_C^U x^C + A_I^U x^I + B_C^U y^C + B_I^U y^I \leq b^U, && \text{(2.24b)}\\
& (y^C, y^I) \in \arg\min_{y^C, y^I} \{ d_C^L y_C + d_I^L y^I : \\
& \qquad A_C^L x^C + A_I^L x^I + B_C^L y^C + B_I^L y^I \leq b^L \}, && \text{(2.24c)}
\end{aligned}$$

where x^C, x^I are the leader's continuous and integer variables respectively, and similarly for the follower's variables y^C, y^I. Unlike MILPs, MILBLPs are $\sum_2^P$-hard. Generally, a problem in the $\sum_k^P$ class can be solved in a non-deterministic polynomial time if there exists some oracle for solving problems in the $\sum_{k-1}^P$ class [104]. This implies that

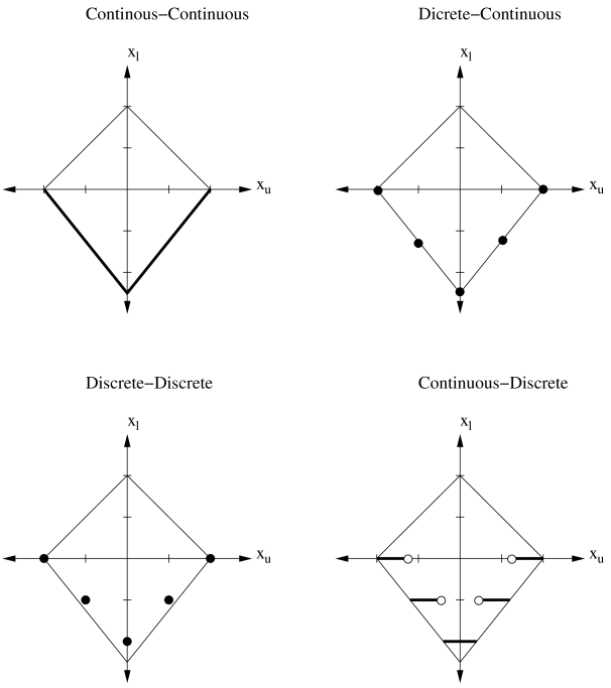


FIGURE 2.5: The Inducible Regions for the four types of variable classifications.

MILBLPs can be solved in a non-deterministic polynomial time if there exists an oracle for solving problems in the $\sum_1^P$ class, which is $\mathcal{NP}$.

As we have presented, it is common practise to reduce the bilevel problem to a single level. When the follower's problem is continuous, cases 1 and 3, we can use the reformuation techniques that have already been described, such as the KKT or strong-duality methods, which is the case in [176]. [165] present exact methods, using branch and bound for the case where the upper levels integer variables are binary. [164] use a simple tabu search, which again only allows for binary leader variables. [65] rewrite the follower's problem using multiparametric programming, from which they obtain the rational reaction set of the follower, which is used to create the single-level formulation.

Generally, integer follower variables lead to a non-convex feasible region and therefore some of the reformulation approaches that rely on gradients and duality cannot be applied. The first set of heuristics developed for solving MILBLPs with continuous and integer leader and follower variables can be found in [113], who use a branch and bound framework with adjusted MILP fathoming rules. Branching is common practise for solving MILPs, where child nodes are created as a result of fractional solutions. With MILPs, bounds can be obtained using the relaxation at the node, which can then be used to fathom unexplored nodes, reducing the computational expense. However, with a BLP, not all fathoming rules for MILPs can be directly applied. As a results, [113] discussed three observations. 1) The solution of the relaxed BLP does not provide a valid bound on the solution of the mixed-integer BLP, 2) solutions to the relaxed BLP in the inducible region cannot in general be fathomed and 3) all integer

solutions to the replaced BLP with some of the followers variables restricted cannot in general be fathomed. [113, 16, 15, 64] present branch and bound algorithms. [15, 64] use a KKT reformulation, by dropping the complementary terms and branching if necessary. [113] present a depth-first approach, whereas [16] present both bredth-and depth-search techniques. As a result of the fathoming rules a branch and bound heuristic may not provide tight bounds.

Similar to branch and bound, branch and cut methods have been used for MILBLP's in [151, 159, 172, 68, 70, 71, 69]. Branch and cut methods for MILBLPs have the same concept as for MILPs and will introduce cuts locally, or globally, to remove any unwanted solutions. [151] generalises the cuts for bilevel problems into three categories: feasibility cuts, optimality cuts and projected optimality cuts.

An early example of a branch and cut algorithm can be found in [48] who use a Chvátal–Gomory cut to remove a solution to the relaxed bilevel problem should the solution not meet the integrality requirements.

[159] call their algorithm the "Watermelon Algorithm" as a result of its resemblance to how one may eat a watermelon, whereby unwanted integer solutions are pips that are not to be eaten and must be removed from the feasible region. Using a branch and bound framework, at every node k of the tree, the HPR, (2.20a)–(2.20c) is solved. The node solution $(x, y)^k$ is then part of a feasibility check to see if it is bilevel feasible. If the point is infeasible, then (2.20d) is violated. Wang and Xu create a polyhedron $\mathcal{C}(t)$, containing the point $(x, y)^k$ and no integer points that are bilevel feasible. They minimise the distance between the facets of this infeasible polyhedron to the facets of the feasible region, given by G^U and G^L. $\mathcal{C}(t)$ is locally removed by partitioning the feasible regions into $m + 1$ sections, where m is the number of facets of $\mathcal{C}(t)$ and m branches are created to be explored further.

The work by [68, 70, 71, 69] is very similar to that in [159, 172]. They too begin by using a branching tree, solving each node problem and creating an infeasible polyhedron if (2.20d) is not satisfied. However, instead of creating m child nodes, they remove a portion of this infeasible region by using an intersection cut, between $\mathcal{C}(t)$ and the feasible region. The results from their tests show that their solver 'consistently outperforms' the alternative methods, even those that exploit problem specific information.

[129, 58] establish that inequalities that remove fractional points remain valid in the bilevel context. Therefore, they employ a 'standard' branch and cut procedure until an integer point is reached. The cuts used in [35] are non-linear and are only applied at points which have been discovered to be bilevel infeasible. Although the cuts are non-linear, they remove more than one bilevel infeasible integer point, unlike [58]. In [58], the inequality used to remove the infeasible solution is

$$\sum_{j \in I} G_j^L(x, y) \leq -1 \tag{2.25}$$

where I is the set of binding inequalities at the infeasible point. As they discuss, the resulting feasibility cut ensures that bilevel infeasible solutions are not generated from their algorithm, however this cut shall only remove a single integer point.

[53, 97] also use a branch and cut framework, using a relaxed value function. Given that the follower's variables are binary and thus bounded, there exists a finite number of solutions available, denoted as $Y = \{y_1, \ldots, y_m\}$, thus formulation (2.21) can be applied. However, even though the number of follower solutions is finite, Y grows exponentiually as the number of lower-level variables increases. Therefore, enumerating all possible solutions would be expensive. The authors choose to construct a relaxed value function by only considering a proportion of Y.

These branch-and-cut frameworks provide a great basis for solving bilevel problems containing integer variables. In Chapter 4 we present our solution methods for a bilevel pricing problem, a majority of which focus on solving the HPR using a branch-and-cut method.

[175] extend the work by [178], by allowing for follower variables to appear in the leaders constraints. Their algorithms are based on a column and constraint generation approach, which iteratively adds integer follower solutions to a relaxed single-level reformulation of the original MILBLP. To create the single level, they use the value function approach similar to [53, 97], by only enforcing optimality with respect to a subset of follower solutions in Y. In contrast to [53, 97], they allow for the leader and follower decision spaces to contain continuous variables. As the follower's integer variables act as parameters, the continuous part of ϕ can be treated as an LP, with the KKT conditions enforcing optimality. As they show, including the KKT conditions for every point in Y can lead to an infeasible single-level problem if the original MILBLP does not satisfy the relatively complete response property, which states that for any (x^C, x^I, y^I) there must exist a feasible and finite y^C for the follower. Rather than restricting themselves to this specific type of problem, they adjust the structure of their single-level problem so that the KKT conditions are only enforced for leaders variables (x^C, x^I) if for $y^I \in Y$ there exists a feasible and finite y^C. This single-level problem is known as the master problem, with two other sub-problems contained within the algorithm. The first of which is a follower optimality check, where for a given (x^C, x^I), the solve the followers problem to get (y^C, y^I). The latter, checks upper-level feasibility for the solution (y^C, y^I), whilst simultaneously searching for follower solutions that have the same follower objective but provide the leader with a better objective. Solution y^I is added to the set Y, unless there exists a solution to the second sub-problem, in which case that solution is added to Y and the algorithm begins the next iteration.

[9] use a multi-parametric approach to partition the original MILBLP into a series of single level ILP's that can be solved in parallel. Their method can also be applied to mixed-integer quadratic bilevel problems, following on from the work by [115, 116] who use McCormick inequalities to linearise any bilinear terms. [62] also use McCormick envelopes to linearise their reformulation. Taking the lower-level problem, they obtain an LP by multiplying the constraints by a polynomial of the follower's variables and linearising using McCormick envelopes. From this single level, they too use multi-parametric programming to obtain numerous integer programs which are solved independently. Although the integer programs are relatively easy to solve, compared to the MILBLPs, the large amount of them can lead to an excessive runtime. [128] recognise this in one of their examples and instead propose a Lagrangian reformulation method to obtain feasible solutions that can be used to generate good bounds and strong cuts.

[126] focus on a specific type of MILBLPs where there exists a single constraint in the upper level which includes follower variables and coefficients are the same as the follower's objective function. They highlight two assumptions, that are present in the applications of the power edge set problem and the minimum zero forcing set problem. A cut generation algorithm is presented for when these assumptions are relevant, along with a more general row and column generation algorithm for when the assumptions do not hold.

[54] transform a specific discrete bilevel problem into a single-level continuous problem using the optimal value reformulation and a penalty function, taken from [106] but originally formulated in [132]. More specifically, the bilevel problem they focus on has a continuous upper level, discrete lower-level, a bilinear lower-level objective but linear with respect to the follower's variables and independent upper and follower feasible regions. They then provide necessary conditions for both the optimistic and pessimistic cases under the assumption that the leader's objective and constraints are continuously differentiable and the follower feasible region is non-empty.

Due to the difficulties encountered with standard MILBLPs, a large proportion of research has been focused towards formulations whose discrete variables are binary. However, we can transform MILBLPs into a binary one by converting the integer variables to a set of binary variables [77], a method also used in [62].

Decomposition methods focus on breaking down the difficult MILBLP into much more manageable sub-problems. Within these sub-problems, there will exist a master sub-problem which will be the focus of the algorithm, with smaller slave sub-problems that will provide useful information, such as cuts, bounds, variables fixings, etc. to be applied to the Master sub-problem. Such a method is heavily associated with Benders Decomposition, see [24] for details. [36, 134, 72] use decomposition approaches to solve MILBLPs that have integer and continuous upper- and lower-level variables respectively. [134] is based on a Benders decomposition method, where cuts are added

to a restricted master problem until convergence to the optimal solution. They use the KKT reformulation approach and active constraints strategy [76] to create a single-level formulation in their sub-problems. [36] use a similar approach, but allow for leader constraints.

As mentioned previously, the majority of methods are restricted to assumptions such as linearity and convexity. Evolutionary methods can be useful for MILBLPs as these do not meet convexity assumptions. [7] use a standard genetic algorithm to solve an interdiction problem based on the electricity grid. [3, 32] focus on the facility location problem, the former using a tabu search matheuristic and the latter applying a genetic algorithm to the single-level problem created after strong duality has been enforced. [37] use a genetic algorithm in both the upper and lower level. Firstly, they decompose the upper level into clusters that span the leaders feasible region, which are evolved in parallel and the best solutions from the clusters are exchanged with a crossover procedure. To evaluate the fitness of the upper-level variables, they solve the lower-level problem, with the leader's variables as parameters, using the same genetic algorithm procedure. [81] tackle the profitable tour problem with a genetic algorithm, where, rather than enumerating the follower's optimal solutions for each leader solution from scratch, they use a subset of possible follower solutions $\mathcal{K}$. For each candidate leader solution, they use knowledge filtering to select the best response from $\mathcal{K}$ along with knowledge assimilation, where they carry out a local search to find a follower response which is a more meaningful estimate. [84] discuss a genetic algorithm that can handle non-linear mixed-integer bilevel problems using exponential and gaussian distributions for the crossover and mutation procedures respectively. [101] present a co-evolutionary algorithm, following the work from [117].

[103] propose a genetic algorithm that solves linear bilevel programming problems, with purely integer variables in the upper and lower level. The general approach for using genetic algorithms with bilevel optimisation is to generate a population of leader feasible solutions and then solve the corresponding follower's problem, using a standard integer programming technique, in this case they use branch and bound. However, the authors recognise that this can be computationally expensive when the size of the population is large. Therefore, they invoke a roulette wheel selection method. Taking the population, they solve the relaxed follower's problem, where integrality is dropped. The fitness of each of these points is given by the leader's objective. Probabilities are then given to each point, with the best fitness having the largest probability and the worst fitness having the smallest probability, with the roulette wheels selection determining which points to solve to optimality.

Chapter 3

Mixed-Integer Linear Bilevel Problems

In this chapter, we introduce the problem which is the focus of this thesis. The problem is based upon a pricing situation, where there exists some set of commodities that can be bought from a market by the leader and the follower. The leader shall purchase commodities from the market at their base cost and re-list them, back on the market, with some taxation added to the cost. The follower's objective is to purchase a set of commodities, which satisfy come combinatorial constraints they may have, in the cheapest manner possible. Figure 3.1 illustrates the problem, along with the order in which the leader and followers actions are taken.

The problem itself is similar to the joint design and pricing model found in [28], which models the leader gaining a revenue by applying a taxation to a subset of arcs in a graph, incurring a cost for any arcs in which a taxation is added, with the follower aiming to traverse between their origin and destination nodes in the cheapest manner. In our case, we assume that the follower's variables are binary and that their feasible region is unaffected by the leader's variables, with the leader's variables only appearing in the follower's objective function. Also, in [28], the arcs that the leader can apply a taxation to are fixed, whereas in our problem they are not.

3.1 Motivation

The original motivation for this problem stemmed from the popular football simulation game FIFA. In FIFA, there exists a game mode where real-life football players are represented by cards. These cards can be used by gamers to create teams to play with against other gamers. To acquire these cards the gamers can purchase them from other

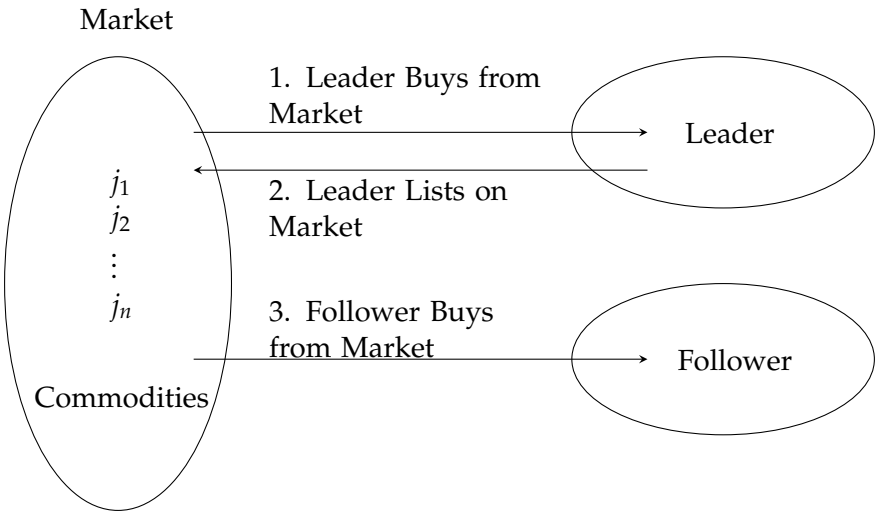


FIGURE 3.1: Outline of the problem to be solved

gamers, who are selling, from the transfer market. The transfer market has an EBAY-like structure, where everyone can buy and sell cards.

One of the game modes is titled squad building challenges, SBC. Here, the gamers will exchange a team of cards for some rewards, with a series of constraints on which cards can be submitted, restricting their choices and thus creating the challenge. Figure 3.2 gives an example of a challenge. Here, the constraints are restricting how many different nationalities and leagues can be used along with the team rating and chemistry, an in-game function which affects the performance of a team.

Gamers try to complete these challenges in the cheapest way possible, making the rewards more appealing. As such, they will try and purchase the cheapest set of players from the transfer market that satisfy the SBC's constraints. These gamers represent the followers in our formulation. Meanwhile, other gamers shall recognise that they can purchase the same players from the transfer market and re-list them at an inflated price and the follower will still purchase them so long that they are part of the cheapest solution. These gamers represent the leader in our formulation.

This problem admits similarities to arbitrage problems, where commodities are instantaneously bought and sold at higher prices to generate profit. Examples of arbitrage can be found in [170, 45, 2, 180], which relate to energy markets and day-ahead prices.

3.2 Single Follower, Unit Supply

In this section, we shall focus on the simplest case, where there exists just a single follower and there a single version of each commodity. The formulation for this is given by

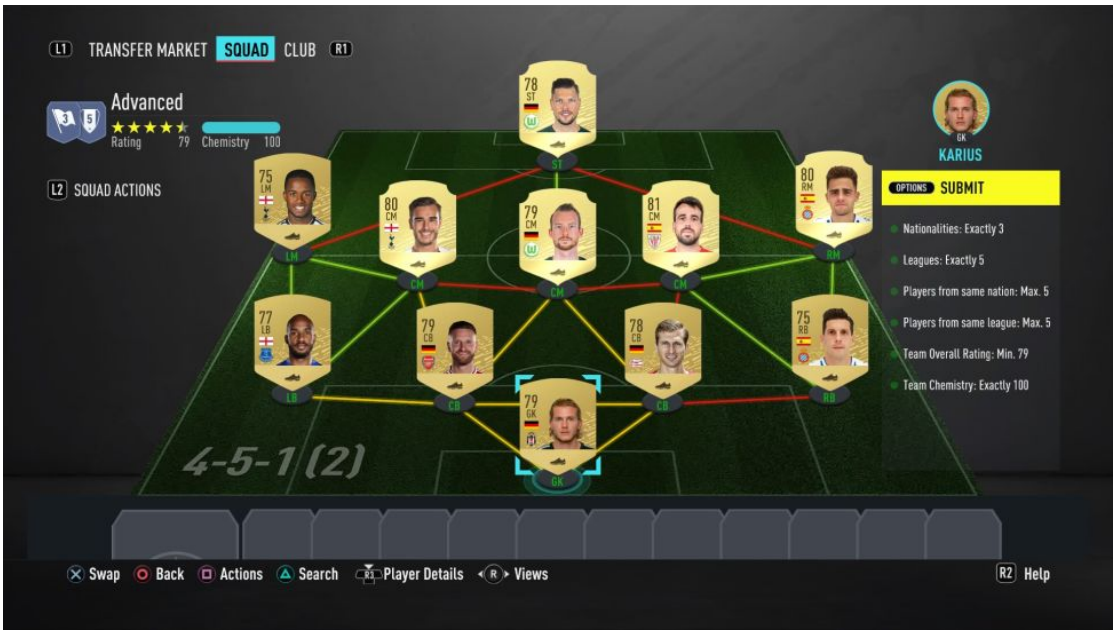


FIGURE 3.2: Example of a team that satisfies an SBC. Here, there are restrictions on how many nationalities and leagues must appear in the team, along with specific values for the teams rating and chemistry.

$$\max_{x,\bar{x}} \quad \sum_{j\in\mathcal{J}} (y_j \bar{x}_j (x_j + c_j) - \bar{x}_j c_j) \tag{3.1a}$$

$$\text{s.t.} \quad x_j \leq M\bar{x}_j \quad \forall j \in \mathcal{J}, \tag{3.1b}$$

$$\sum_{j\in\mathcal{J}} \bar{x}_j c_j \leq B, \tag{3.1c}$$

$$x \in [0, M]^n, \tag{3.1d}$$

$$\bar{x} \in \{0,1\}^n, \tag{3.1e}$$

$$y \in \arg\min_{y} \{ \sum_{j\in\mathcal{J}} y_j (x_j + c_j), \tag{3.1f}$$

$$Ay^\top \leq b, \tag{3.1g}$$

$$y \in \{0,1\}^n \}. \tag{3.1h}$$

Here, x and $\bar{x}$ are the leader's variables, with x_j being the taxation applied to commodity j, where $\mathcal{J}$ is the index set of the commodities, $|\mathcal{J}| = n$, and $\bar{x}_j$ is the leaders binary decision as to whether the leader should buy commodity j, to allow them to apply a taxation to generate profit. y are the follower's variables, where y_j is a binary variable as to whether they purchase commodity j. c_j is the base cost for each commodity j, M is the maximum taxation that the leader can apply and B represents the budget of the leader.

The leader's objective, (3.1a), has two parts, $y_j \bar{x}_j (x_j + c_j)$ and $-\bar{x}_j c_j$, which represent the

leader's income and expenditure, respectively. The leaders income on a commodity j is equal to the base cost, c_j, plus the taxation, x_j, that they have applied. However, they shall only receive this if they have purchased the commodity, i.e $\bar{x}_j = 1$, and if the follower also purchases this commodity, i.e. $y_j = 1$, resulting in $y_j\bar{x}_j(x_j + c_j)$. This can be reduced to $y_j(x_j + \bar{x}_j c_j)$ as when $\bar{x}_j = 0$ then $x_j = 0$, due to constraint (3.1b), thus the quadratic $y_j x_j$ is equal to the cubic $y_j\bar{x}_j x_j$. The leaders expenditure is the sum of the commodities that they have bought at their initial cost, giving $\sum_{j\in\mathcal{J}} \bar{x}_j c_j$. Constraint (3.1b) is an implied upper bound constraint and ensures that the leader cannot apply any taxation to a commodity that they have not bought. (3.1c) is the leaders budget constraint. (3.1d) limits the taxation range, only allowing non-negative taxation. (3.1f) is the follower's objective function and is the product of the follower's decision variable and the final price of the commodities, after any taxation from the leader is added and (3.1g) is the set of combinatorial constraints that limit the choice of the follower.

Comparing this formulation to the generic bilevel formulation (2.1), we can see that the constraint regions for the leader and follower are independent. The leader is only constrained by their own budget, as to which commodities they can purchase and the follower is only constrained by their own combinatorial constraints. This means that for any point $(x, \bar{x})$ generated by the leader, there will always exist a solution y that the follower can respond with, assuming that the follower's feasible region is non-empty. This feature shall become useful later, when we introduce a solution method that utilises cutting planes.

In similar bilevel problems, an assumption is made in that there exists some non-taxable commodities, always available for the follower to purchase. The purpose of this is to stop the leader just assigning unlimited taxations, causing infinite objectives for both the leader and follower. In our case, there are no commodities which are unavailable to the leader, but we have prevented the infinite objective in two ways. First, the leader has a budget constraint given by (3.1c). In most realistic cases, this means that the leader will not be able to purchase every commodity, so there should exist some commodities which have no taxation. Second, and possibly most importantly, we have capped the taxation that can be applied to any commodity to M. This means that the leaders objective function is upper bounded to Mn, which can only be achieved if the leader buys every commodity, applied the maximum taxation and the follower then buys every commodity. Likewise, the follower's objective has an upper bound of $\min\{y_j(M + c_j) : Ay^\top \leq b, y \in \{0,1\}^n\}$ which is clearly finite given that y is a binary variable and n is also finite.

In some bilevel problems, we have to state whether we are considering the optimistic or the pessimistic case, when the follower can react in more than one way, each producing the same follower objective, but differing leader objectives. In our case, we assume the optimistic case, however, we can also show that, similar to many other pessimistic

problems, such as the one in section 2.2 of [17], should we assume pessimism, the maximum objective turns into a supremum. This supremum is thus a value which the leader can always improve towards but never reach. As a result of the continuous taxation, in the pessimistic case, should there be multiple solutions producing an equal objective for the follower, the leader can reduce the total taxation being applied to the one that gives the leader a better objective function by some ϵ. This now leaves the follower with a single optimal response and the leader will have only lost ϵ from their objective function. For the case where the taxation can only take integer values, this is clearly not the case.

As mentioned, the leaders objective function has two parts, their income $y_j\bar{x}_j(x_j + c_j)$ and their expenditure $-\bar{x}_j c_j$. The leader shall only receive income for a commodity j when $\bar{x}_j = 1$ and $y_j = 1$, i.e., both the leader and follower have purchased j. Therefore, one may assume that in the optimal solution there will not exist some j such that $\bar{x}_j = 1$ and $y_j = 0$, however Example 3.1 illustrates that this is not the case.

Example 3.1. *Lets assume that there are 2 commodities, j_1 and j_2 with costs of c_{j_1} and c_{j_2} respectively, with $c_{j_1} < c_{j_2}$. We also assume that the leader's budget constraint allows for the leader to purchase both commodities, i.e. $B \geq c_{j_1} + c_{j_2}$ and that the only follower constraint is that they must purchase at least one of the commodities. If the leader purchases j_2 only, then, because taxation is only positive, the follower will always react by purchasing j_1, giving the leader an objective value of $0 - c_2 = -c_2$. If the leader purchases commodity j_1 only, then the maximum taxation they can apply is $c_{j_2} - c_{j_1}$, as any more than this and the follower will respond by purchasing j_2, which gives the leader an objective value of $((c_{j_2} - c_{j_1}) + c_{j_1}) - c_{j_1} = (c_{j_2} - c_{j_1})$. However, if the leader purchases both j_1 and j_2, the leader can keep increasing the taxation on j_1 to M as they are able to also increase the price of j_2, such that j_1 always remains the cheaper option. So, if the leader sets $x_{j_1} = M$ and $x_{j_2} \geq M - (c_{j_2} - c_{j_1})$, j_1 shall remain the cheapest commodity and the follower will purchase it, with the leader getting an objective value of $(M + c_{j_1}) - (c_{j_1} + c_{j_2}) = M - c_{j_2}$, which is the optimal solution even though we have $\bar{x}_{j_2} = 1$ and $y_{j_2} = 0$.*

We also obtain the following observations.

Observation 3.1. *For any instance of* (3.1)*, the leaders optimal objective function is lower bounded by 0.*

Proof. For any instance, the leader has the option to do nothing, i.e. $x = \bar{x} = \{0\}^n$. This means that they have 0 expenditure, because they have not bought anything and they have no income because they have nothing to apply a taxation to. Therefore, they can always achieve an objective of 0 and will never have a negative objective. □

Observation 3.2. *For a given solution $(x, \bar{x}, y)$, if there exists some $j \in \mathcal{J}$ such that $\bar{x}_j = 1$ and $x_j = 0$, then the leader can improve their objective by c_j by setting $\bar{x}_j = 0$.*

Proof. If there exists some $j \in \mathcal{J}$ such that $\bar{x}_j = 1$ and $x_j = 0$, then the leader has purchased a commodity but has not applied any taxation to it, thus the commodities price has remained unchanged. Therefore, by not buying it, the final cost of the followers solutions remain unchanged, so their response remains the same and the leader has c_j less expenditure. □

Observation 3.3. *For the formulation* (3.1)*, the optimal objective value for the leader is the same for when we allow the leader to apply negative taxation, i.e.* $x \in [-M, M]$.

Proof. Let's assume that in (3.1) we have replaced constraint (3.1d) with $x \in [-M, M]$ and that the optimal solution is $(x^*, \bar{x}^*, y^*)$, where we define $X^- = \{j \in \mathcal{J} : x_j^* < 0\}$ as the set of indices where negative taxation has been applied. First, if for all $j \in \mathcal{J}$, such that $y_j^* = 1$, we have $x_j^* \geq 0$, then clearly we can place 0 as a lower bound on x. Secondly, if there exists some $j \in X^-$, such that $y_j^* = 0$, then this is not an optimal solution. The leader has bought a commodity and then applied a negative taxation, which does the opposite to that described in Example 3.1, which means they can improve their objective by $\sum_{j \in X^-} c_j$ by not purchasing the commodities in X^-.

Lastly, if there exists some $j \in X^-$, such that $y_j^* = 1$, then there exists an alternative solution $(x', \bar{x}', y^*)$, which gives the leader the same objective function, but for all $j \in \mathcal{J}$, $x'_j \geq 0$. As already shown, we have a lower bound of 0 on the leaders optimal objective.

Thus, by defining $X^+ = \{j \in \mathcal{J} : x_j^* > 0\}$, we must have that $\sum_{j \in X^+} y_j^* x_j^* \geq \sum_{j \in X^-} |y_j^* x_j^*|$, i.e., the amount of income the leaders receive from the positive taxation must be greater than the absolute of the negative taxations. If this is the case, then, by defining x' and $\bar{x}'$ such that

$$x'_j = \begin{cases} 0, & j \in X^-, \\ x_j^* - \dfrac{\sum_{j \in X^-} |x_j^*|}{|X^-|}, & j \in X^+, \end{cases} \tag{3.2a}$$

$$\bar{x}' = \begin{cases} 0, & j \in X^-, \\ 1, & j \in X^+, \end{cases} \tag{3.2b}$$

the leader should achieve a better objective as, although the amount of income from taxation remains the same, they have less expenditure with $\bar{x}'$ than with $\bar{x}^*$. □

3.3 Multiple Followers

In (3.1), we assumed that there was a single follower, with a single set of constraints. Now, we shall expand the formulation by assuming there exists multiple followers, that

can be partitioned into groups, each with their own unique set of constraints to satisfy. As such, we get the following formulation

$$\max_{x,\bar{x}} \quad \sum_{j\in\mathcal{J}} \Big(\sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} y_j^{ik}(x_j + \bar{x}_j c_j) - \bar{x}_j c_j \Big) \tag{3.3a}$$

$$\text{s.t.} \quad x_j \leq M\bar{x}_j \quad \forall j \in \mathcal{J}, \tag{3.3b}$$

$$\sum_{j\in\mathcal{J}} \bar{x}_j c_j \leq B, \tag{3.3c}$$

$$x \in [0, M]^n, \tag{3.3d}$$

$$\bar{x} \in \{0,1\}^n, \tag{3.3e}$$

$$y \in \arg\min_y \{ \sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} \sum_{j\in\mathcal{J}} y_j^{ik}(x_j + c_j) \tag{3.3f}$$

$$A^i y^{ik} \leq b^i \quad \forall i \in \mathcal{I}, \forall k \in \{1, \ldots, d^i\}, \tag{3.3g}$$

$$\sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} y_j^{ik} \leq 1 \quad \forall j \in \mathcal{J}, \tag{3.3h}$$

$$y^{ik} \in \{0,1\}^n \}. \tag{3.3i}$$

The constraints for the leader remain unchanged, with just the objective function being altered to reflect that income can come from any one of the followers from each group. $\mathcal{I}$ is the index set of groups that the followers can be partitioned into, with $|\mathcal{I}| = m$, and d^i is the number of followers that belong to group i. y^{ik} is a binary vector, of length n, that represents the response of the k$^{\text{th}}$ follower in group i. For example, $y_j^{ik} = 1$ if and only if the k$^{\text{th}}$ follower from the i$^{\text{th}}$ group purchases commodity j. The objective function (3.3f) is now the original cost, plus any taxation applied by the leader, for each commodity across every follower within every group. (3.3g) represents the combinatorial constraints that limit the follower response, with A^i and b^i being the constraints for group i respectively. (3.3h) ensures that each commodity can only be selected at most once across all followers.

With the introduction of multiple followers, we assume that they act collectively as one and do not compete for commodities. Should the followers be competing for commodities, there would be an order to their reactions, which can cause the sum of the followers reaction to be greater than if they were working collectively.

Example 3.2. *Let $i = 2$ and $d^1 = d^2 = 1$, i.e., two followers with differing constraints. Assume there are three commodities, A, B and C, whose prices are given by $(2, 1, 100)$ and the feasible responses for follower one and two are $\{A, B\}$ and $\{B, C\}$ respectively. Should we use* (3.3) *and the followers work together, then their responses would be A and B respectively which gives a total cost to the followers of* 3*. However, should follower one respond first, then they*

TABLE 3.1: Leader solutions for Example 3.3

	$x \in [0, M]^3$		$x \in [-M, M]^3$	
$\bar{x}$	x	F^U	x	F^U
$(0,0,0)$	$(0,0,0)$	0	$(0,0,0)$	0
$(1,0,0)$	$(1,0,0)$	1	$(1,0,0)$	1
$(1,1,0)$	$(M,M,0)$	$M-1.5$	$(M,M,0)$	$M-1.5$
$(1,0,1)$	$(M,0,M)$	$M-1$	$(M,0,M)$	$M-1$
$(0,1,0)$	$(0,0,0)$	-1.5	$(0,0,0)$	-2
$(0,1,1)$	$(0,M,M)$	$M-1.5$	$(0,-0.5,M)$	$M-0.5$
$(0,0,1)$	$(0,0,1)$	1	$(0,0,1)$	1

would react by purchasing B, as $1 < 2$*, leaving the second follower to purchase C, giving a total cost to the followers of* 101*.*

By allowing this order in the followers reactions, we have discovered a Stakelberg solution, where neither follower wants to, or can in the second followers case, deviate from their reaction as it would give them a worse payoff, individually. However, collectively they could have achieved a much smaller cost, commonly referred to as the social optimal solution. Along with this, should the followers respond in an order then Observation 3.3 no longer holds, as outlined by Example 3.3

Example 3.3. *Similar to before, let* $i = 2$ *and* $d^1 = d^2 = 1$*. Let there be three commodities,* (A, B, C)*, whose prices are given by* $(1, 1.5, 1)$*, the feasible responses of the followers are* $\{A, B\}$ *and* $\{B, C\}$ *respectively, with the first follower reacting first and the leader has a budget of 3. As such, the leader has seven possible ways they can purchase the commodities, given by Table 3.1. As we can see, in six out of the seven possible choices for* $\bar{x}$*, the leader would act the same irrespective of if negative taxation is allowed. However, when* $\bar{x} = (0, 1, 1)$ *this is not the case. When* $x \in [0, M]^3$*, as the base cost of commodity B is greater than that of commodity A, the first follower shall always react by purchasing commodity A. Thus the second follower can choose between the cheapest of commodity B or C. Therefore, as the leader has purchased both of these commodities, they can apply the maximum taxation to both and the second follower will still have to purchase commodity C. Hence, the leader shall receive* $(M + 1)$ *from the second follower, have a total expenditure of* 2.5*, giving them a final objective of* $M - 1.5$*.*

However, the leader can achieve a greater objective by introducing negative taxations. When $x \in [-M, M]^3$ *and* $\bar{x} = (0, 1, 1)$*, the optimal solution for the leader is* $x = (0, -0.5, M)$*. By applying the negative taxation to B, both A and B have an equal cost for the first follower, thus they can choose either commodity. But, given that we are assuming the optimistic case, they shall select B (this can also be achieved by setting the taxation to* $-(1 + \epsilon)$*, for some small* ϵ*). As the first follower has responded by purchasing B, the second follower is only left to purchase C, thus the leader can apply the maximum taxation. Overall, this means that the leaders income is* $(-0.5 + 1.5) + (M + 1)$ *and their expenditure is* 2.5*, giving a final objective of* $M - 0.5$*.*

In Example 3.3 we find that $\bar{x} = (0, 1, 1)$ and $\bar{x} = (1, 0, 1)$ are similar, in that they are trying to force the first follower to select commodity B such that the second follower is left with commodity C. When $\bar{x} = (1, 0, 1)$, the leader shall inflate the price of A to make B appealing whereas in $\bar{x} = (0, 1, 1)$ the leader reduces the price of B. When the prices of the commodities are set to $(1, 1.5, 1)$ we can achieve a better objective with $x \in [-M, M]$ because the amount of negative taxation is less than the cost of A, i.e. $0.5 < 1$. Should the prices of the commodities have been $(1, 2.5, 1)$, and the leaders budget been 3.5, then $\bar{x} = (0, 1, 1)$ and $\bar{x} = (1, 0, 1)$ would have optimal objective function values of $M - 1.5$ and $M - 1$ respectively. Thus, the leader would not want to apply a negative taxation because the negative taxation they would have to apply to B is greater than the cost of A, i.e. $1.5 > 1$.

For the remainder of this thesis, we shall assume that the followers are trying to achieve a socially optimal solution and work collectively. Real-world examples of this can arise where we have a single follower with numerous separate combinatorial constraints they must satisfy. If the constraints are independent we can treat each as a separate follower and the objective will be to find the socially optimal solution. As such, we shall also assume that only positive taxation is allowed.

We have also opened up the possibility for symmetric solutions. These solutions occur, when variables can be interchanged, without changing the structure of the problem [74]. In this case, we can see that the followers for each group can be arranged in any order and the problem remains the same. Likewise, for any follower solution, the way in which commodities are distributed amongst the followers has no impact on the leader, as the leader is only interested in the fact that commodities are bought from them.

3.4 Non-Unit Supply

We shall now assume that each commodity is no longer unique and that there can exist multiple duplicates of the same commodity. By increasing the supply, the structure of the bilevel problem has been altered. Previously, because there was a single version of each commodity, if the leader had bought a commodity that the follower also wanted, then the follower would have to pay the taxation applied. However, now there are duplicate commodities, if the follower wants a commodity, they can decide between ones that have been bought by the leader and ones that have not. Clearly, as there is only positive taxation being applied, the follower will always get commodities the leader has not bought before purchasing from the leader. Thus, the bilevel problem now becomes

$$\max_{x,\bar{x}} \quad \sum_{j\in\mathcal{J}} \Big(\max\{0, \sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} y_j^{ik} - (s_j - \bar{x}_j)\}(x_j + c_j) - \bar{x}_j c_j \Big) \tag{3.4a}$$

$$\text{s.t.} \quad x_j \leq M\bar{x}_j \quad \forall j \in \mathcal{J}, \tag{3.4b}$$

$$\sum_{j\in\mathcal{J}} \bar{x}_j c_j \leq B, \tag{3.4c}$$

$$\bar{x}_j \leq s_j \quad \forall j \in \mathcal{J}, \tag{3.4d}$$

$$x \in [0, M]^n, \tag{3.4e}$$

$$\bar{x} \in \mathbb{Z}^n_{\geq 0}, \tag{3.4f}$$

$$y \in \arg\min_y \{ \sum_{j\in\mathcal{J}} \Big(\max\{0, \sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} y_j^{ik} - (s_j - \bar{x}_j)\} x_j + \sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} y_j^{ik} c_j \Big) : \tag{3.4g}$$

$$A^i y^{ik} \leq b^i \quad \forall i \in \mathcal{I}, \forall k \in \{1, \dots, d^i\}, \tag{3.4h}$$

$$\sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} y_j^{ik} \leq s_j \quad \forall j \in \mathcal{J}, \tag{3.4i}$$

$$y^{ik} \in \{0, 1\}^n \}. \tag{3.4j}$$

The max operator, in the leader and follower objectives, ensure that the follower will only purchase from the leader if they must. $s_j - \bar{x}_j$ is the amount of commodity j that has not been bought by the leader. Therefore, if $\sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} y_j^{ik} \leq (s_j - \bar{x}_j)$, then the follower does not need to purchase j from the leader. However, if $\sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} y_j^{ik} \geq (s_j - \bar{x}_j)$, the commodities not bought by the leader do not fulfil the followers needs, therefore the follower must purchase from the leader and pay the taxation applied. This max operation could have been included in the single supply case, however, we can see that if $s_j = 1$, then the max operation can only take the value 1 when $y_j^{ik} = \bar{x}_j = 1$ and is 0 otherwise, thus the max operation can be represented by $y_j^{ik}\bar{x}_j$.

As the leader can now purchase multiple versions of each commodity, then $\bar{x}$ must go from a binary variable to an integer one, whose upper bound is the supply for each commodity (3.4d). Likewise, the number of each commodity j that the followers purchase, must have an upper bound equal to the supply for that commodity, (3.4i).

The max operator is one which we cannot use directly with most state-of-the-art solvers. Therefore, we shall present two further formulations, which we shall call the dichotomic and max value formulations.

In these formulations, we remove the max operator and introduce additional variables and constraints. In both dichotomic and max value formulations, the new variables are considered to be follower variables and the constraints are placed in the followers problem only. The max operator is a representation of how the follower reacts given the commodities that the leader has purchased, i.e. the follower can only purchase

commodities from the leader if the leader has already bought them. Thus, given that the max operator is used to represent the followers reaction, any variables and constraints used to replace the max operator are considered to be follower variables and constraints.

3.5 Dichotomic Formulation

In (3.4), we find a max operation in both the leaders and followers objective function, where this determines the quantity of each commodity that the follower chooses to buy from the leader. Given that the max operation arises because we have to distinguish between commodities that the follower buys from the leader and not from the leader, we shall introduce an additional set of variables that does exactly this. Let $\bar{y}$ be the follower variable for commodities bought from the leader and y for commodities that are not bought from the leader. The bilevel formulation then becomes

$$\max_{x,\bar{x}} \quad \sum_{j\in\mathcal{J}} \Big(\sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} \bar{y}_j^{ik}(x_j + c_j) - \bar{x}_j c_j \Big) \tag{3.5a}$$

$$\text{s.t.} \quad (3.4b) - (3.4f), \tag{3.5b}$$

$$(y, \bar{y}) \in \arg\min_{y,\bar{y}} \{ \sum_{j\in\mathcal{J}} \sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} \bar{y}_j^{ik}(x_j + c_j) + y_j^{ik} c_j : \tag{3.5c}$$

$$A^i(\bar{y}^{ik} + y^{ik}) \leq b^i \quad \forall i \in \mathcal{I}, \forall k \in \{1, \dots, d^i\}, \tag{3.5d}$$

$$\sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} (\bar{y}_j^{ik} + y_j^{ik}) \leq s_j \quad \forall j \in \mathcal{J}, \tag{3.5e}$$

$$\sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} \bar{y}_j^{ik} \leq \bar{x}_j \quad \forall j \in \mathcal{J}, \tag{3.5f}$$

$$\sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} y_j^{ik} \leq s_j - \bar{x}_j \quad \forall j \in \mathcal{J}, \tag{3.5g}$$

$$\bar{y}_j^{ik} + y_j^{ik} \leq 1 \quad \forall j \in \mathcal{J}, \forall i \in \mathcal{I}, \forall k \in \{1, \dots, d^i\}, \tag{3.5h}$$

$$\bar{y}^{ik}, y^{ik} \in \{0,1\}^n \}. \tag{3.5i}$$

As we can see, the leader's objective function is still in two parts, the income and expenditure. However, the income is purely determined by the final price of commodities that the follower decides to purchase from the leader. Constraints (3.5d) and (3.5e) are the same as before, but are adjusted slightly to account for y being spread over two variables now. Constraint (3.5f) limits the number of commodities that the follower assigns to purchasing from the leader by the amount that the leader has bought. (3.5g) limits the number of commodities the followers doesn't purchase from the leader to

the amount that the leader has not purchased. Lastly, constraint (3.5h) ensures that a follower cannot purchase the same commodity from both the leader and the market.

By introducing $\bar{y}$, we have dissolved the max operator. However, now the follower's feasible region depends on the leader's variable $\bar{x}$ by constraints (3.5f) and (3.5g). This now means that the feasible region for the follower is perturbed by the leaders variables, which can cause complications in the solution methods presented later. Note, a set of commodities shall always provide a feasible solution for the follower. The dependency only affects whether the follower purchases these commodities from the leader or the market.

Given that the max operator was in the leaders objective, then one may assume that the constraints (3.5f),(3.5g) should also be leader constraints. However, we have remodelled the max operator as a follower variable, $\bar{y}$, which is dependent on a leader variable, $\bar{x}$. Therefore, the corresponding constraints belong in the followers problem. Likewise, placing these constraints in the leaders problem would be counter-intuitive with respect to how we have modelled our problem. Should these constraints be in the leaders problem, we are implying that the reaction of the follower should directly determine the quantity of each commodity that the leader should purchase. However, in Figure 3.1, we state the the leader purchases commodities before the follower, thus cannot be directly influenced by the followers reaction.

3.6 Max Value Formulation

Another way to handle the max operator is to introduce two additional variables, z and γ, where z shall take the value of the max operation and γ shall be a binary variable that determines whether $z = 0$ or $z = \sum_{i\in\mathcal{I}}\sum_{k=1}^{d^i} y_j^{ik} - (s_j - \bar{x}_j)$. We define γ in the following way

$$\gamma_j = 1 \implies \sum_{i\in\mathcal{I}}\sum_{k=1}^{d^i} y_j^{ik} - (s_j - \bar{x}_j) \geq 0, \tag{3.6a}$$

$$\gamma_j = 0 \implies \sum_{i\in\mathcal{I}}\sum_{k=1}^{d^i} y_j^{ik} - (s_j - \bar{x}_j) \leq 0, \tag{3.6b}$$

for all j. (3.6a) and (3.6a) imply that when $\gamma = 1$, the followers purchase more commodities than the amount the leader has not bought and therefore must purchase some commodities from the leader. This can be achieved linearly using the Big-M constraints

$$\bar{M}(1-\gamma_j) \geq -\Big(\sum_{i\in\mathcal{I}}\sum_{k=1}^{d^i} y_j^{ik} - (s_j - \bar{x}_j)\Big), \tag{3.7a}$$

$$\bar{M}\gamma_j \geq \sum_{i\in\mathcal{I}}\sum_{k=1}^{d^i} y_j^{ik} - (s_j - \bar{x}_j), \tag{3.7b}$$

for all $j \in \mathcal{J}$, with a sufficiently large $\bar{M}$, which in this case can be set to s_j. When $\sum_{i\in\mathcal{I}}\sum_{k=1}^{d^i} y_j^{ik} > s_j - \bar{x}_j$, (3.7b) ensures that $\gamma = 1$, when $\sum_{i\in\mathcal{I}}\sum_{k=1}^{d^i} y_j^{ik} < s_j - \bar{x}_j$, (3.7a) enforces $\gamma = 0$ and when $\sum_{i\in\mathcal{I}}\sum_{k=1}^{d^i} y_j^{ik} = s_j - \bar{x}_j$, γ can take either value, which is not an issue because in this case we have $\max\{0,0\}$, whose result is always 0. We then define $z_j = \max\{0, \sum_{i\in\mathcal{I}}\sum_{k=1}^{d^i} y_j^{ik} - (s_j - \bar{x}_j)\}$ by introducing the constraints

$$z_j \geq 0, \tag{3.8a}$$

$$z_j \geq \sum_{i\in\mathcal{I}}\sum_{k=1}^{d^i} y_j^{ik} - (s_j - \bar{x}_j), \tag{3.8b}$$

$$z_j \leq s_j\gamma_j, \tag{3.8c}$$

$$z_j \leq \sum_{i\in\mathcal{I}}\sum_{k=1}^{d^i} y_j^{ik} - (s_j - \bar{x}_j) + s_j(1-\gamma_j), \tag{3.8d}$$

for all $j \in \mathcal{J}$, which enforce $z_j = \sum_{i\in\mathcal{I}}\sum_{k=1}^{d^i} y_j^{ik} - (s_j - \bar{x}_j)$ when $\gamma_j = 1$ and $z_j = 0$ when $\gamma_j = 0$. As discussed, when $\sum_{i\in\mathcal{I}}\sum_{k=1}^{d^i} y_j^{ik} = s_j - \bar{x}_j$, although γ_j can be either 0 or 1, both values produce $z_j = 0$. The bilevel formulation then becomes

$$\max_{x,\bar{x},z} \quad \sum_{j\in\mathcal{J}} z_j(x_j + c_j) - \bar{x}_j c_j \tag{3.9a}$$

$$\text{s.t.} \quad (3.4b) - (3.4f), \tag{3.9b}$$

$$z \in \arg\min_{y,\gamma,z}\{\sum_{j\in\mathcal{J}} z_j x_j + \sum_{i\in\mathcal{I}}\sum_{k=1}^{d^i} y_j^{ik} c_j : \tag{3.9c}$$

$$(3.4h) - (3.4j), \tag{3.9d}$$

$$(3.7), (3.8) \quad \forall j \in \mathcal{J}, \tag{3.9e}$$

$$z_j \leq s_j \quad \forall j \in \mathcal{J}, \tag{3.9f}$$

$$\gamma \in \{0,1\}^n, \tag{3.9g}$$

$$z \in \mathbb{Z}^n_{\geq 0}\}. \tag{3.9h}$$

One may assume that because the follower is minimising then we do not need the upper bound (3.8d). However, although removing this constraint would not affect the follower's objective, it will affect the leader's. As we can see, in the leader's objective we have the product $z_j x_j$ and $z_j c_j$ and in the follower's objective we just have $z_j x_j$. Thus, when $x_j = 0$ only the $z_j c_j$ term can be non-zero. Therefore, by not applying the upper bound (3.8d) the leaders objective can be increased giving an incorrect objective.

3.7 Linearisation

Thus far, the formulations that have been created have included non-linear elements, which can provide difficulty in solving. Therefore we shall present a linearisation method based on the McCormick envelopes.

First described in [109] and shown to be envelopes in [4], the McCormick constraints can relax a bi-linear program. Given two variables $x_1, x_2 \in \mathbb{R}$ such that $x_i \in [l_i, u_i]$ for $i = 1, 2$, the McCormick constraints of the bilinear product $x_3 = x_1 x_2$ are given by

$$x_3 \geq l_1 x_2 + l_2 x_1 - l_1 l_2, \tag{3.10a}$$
$$x_3 \geq u_1 x_2 + u_2 x_1 - u_1 u_2, \tag{3.10b}$$
$$x_3 \leq u_1 x_2 + l_2 x_1 - u_1 l_2, \tag{3.10c}$$
$$x_3 \leq l_1 x_2 + u_2 x_1 - l_1 u_2. \tag{3.10d}$$

These constraints work as pairs, providing over- and under-estimations for x_3. An advantages that comes with the McCormick constraints is that should either one of x_1 or x_2 be binary, then the relaxation becomes exact. For example, if $x_1 \in \{0, 1\}$ then the constraints (3.10) become $x_3 \geq 0$, $x_3 \geq x_2 - u_2$, $x_3 \leq x_2 - l_2$ and $x_3 \leq 0$. When $x_1 = 0$, these can only be satisfied with $x_3 = 0$. Likewise, when $x_1 = 1$, the constraints become $x_3 \geq l_2$, $x_3 \geq x_2$, $x_3 \leq x_2$ and $x_3 \leq u_2$, which can only be satisfied with $x_3 = x_2$. For ease of writing, we shall define the set of McCormick constraints, given by (3.10), by $\mathcal{MC}(x_3, x_1, x_2)$, where we define $x_3 = x_1 x_2$.

For situations where one of the variables takes an integer value, the McCormick constraints are not exact and provide a relaxation. However, by performing a binary expansion on the integer variable, we can transform the single non-linear term in many non-linear terms that include a binary variable, for which the McCormick envelopes shall be exact. As before, lets take the product $x_3 = x_1 x_2$ and assume that $x_1 \in \mathbb{Z}$. We

shall define our binary expansion, often referred to as a *logarithmic*-binarisation, of x_1 as

$$x_1 = \sum_{i=1}^{\lfloor \log_2 u_1 \rfloor + 1} 2^{i-1}\alpha_i \quad \alpha_i \in \{0,1\} \quad \forall i \in [1, \lfloor \log_2 u_1 \rfloor + 1]. \tag{3.11a}$$

Thus, the non-linear formulation becomes linear by substituting x_3 for any $x_1 x_2$ and introducing the constraints

$$x_1 = \sum_{i=1}^{\lfloor \log_2 u_1 \rfloor + 1} 2^{i-1}\alpha_i, \tag{3.12a}$$

$$\alpha_i \in \{0,1\} \quad \forall i \in [1, \lfloor \log_2 u_1 \rfloor + 1], \tag{3.12b}$$

$$\mathcal{MC}(\beta_i, \alpha_i, x_2) \quad \forall i \in \{1, \lfloor \log_2 u_1 \rfloor + 1\}, \tag{3.12c}$$

$$x_3 = \sum_{i=1}^{\lfloor \log_2 u_1 \rfloor + 1} 2^{i-1}\beta_i. \tag{3.12d}$$

Alternative binarisation techniques can be found in [46]. Firstly, the *full*-binarisation, which is given by

$$x_1 = \sum_{i=1}^{u_1} i z_i, \tag{3.13a}$$

$$\sum_{i=1}^{u_1} z_i \leq 1, \tag{3.13b}$$

$$z_i \in \{0,1\} \quad \forall i \in [1, u_1], \tag{3.13c}$$

and, secondly, the *unary*-binarisation is given by

$$x_1 = \sum_{i=1}^{u_1} z_i, \tag{3.14a}$$

$$1 \geq z_1 \geq z_2 \geq \cdots \geq z_{u_1} \geq 0, \tag{3.14b}$$

$$z_i \in \{0,1\} \quad \forall i \in [1, u_1]. \tag{3.14c}$$

[46] briefly mention these binarisation techniques, explaining how the *full* technique is studied in [138], [6] and [78], *unary* is discussed in [133] and [25] and *logarithmic* can

be found in [121] and [78]. However, [46] conclude by discussing the lack of computational results which compare the performances of these binarisation techniques.

3.7.1 A note on the results of Dempe & Kue

[53] focus their attention to a specific MILBLP, which can be reformulated as the following single-level formulation after using the value function approach;

$$\max_{x} \quad d_1^\top x + d_2^\top y \tag{3.15a}$$
$$\text{s.t.} \quad Dx \leq d, \tag{3.15b}$$
$$Ay \leq b, \tag{3.15c}$$
$$x, y \in \{0,1\}^n, \tag{3.15d}$$
$$x^\top y \leq x^\top y^i \quad \forall y^i \in \mathcal{Y}, \tag{3.15e}$$

where $\mathcal{Y}$ is the finite set of all feasible solutions for the follower. Very similar to our unit supply problem, (3.15) has a bi-linear follower objective $x^\top y$, which is linear w.r.t. the followers variables and a follower feasible region $Ay \leq b$, which is not perturbed by the leader's variables.

They replace the non-linear value functions constraints (3.15e) with a relaxed linear constraint. The advantage of doing so means that no additional variables or constraints need to be added to the problem, unlike the McCormick approach. However, they have increased the size of the convex hull, which now contains binary points that are not feasible solutions to the origin bilevel problem. In what follows, we will show how, by using a McCormick reformulation, we can generate a tighter feasible region than that in [53]. To begin with, we define the constraints in their relaxation as follows. Let X and Y be the indices for variables x and y, respectively. Let $N^+ = \{1, \ldots, n\}$, $L_i = \{j \in Y : y_j^i = 1\}$ and $N_i = N^+ \cup L_i$. A set $M \subset N_i$ is defined to be a cover of the i$^{\text{th}}$ value function constraint if $|M| > |L_i|$. M can be partitioned into two sets, M_{N^+} and M_{L_i}, corresponding to the indices in M that come from the sets N^+ and L_i, respectively. Note, if M is a cover for the i$^{\text{th}}$ value function, then the set M_{N^+} cannot be empty. Using these definitions, [53] produce the following lemma

Lemma 3.4. *If $x^\top y \leq x^\top y^i$ holds, then*

$$\sum_{j \in M_{N^+}} (1 - x_j) + \sum_{j \in M_{N^+}} (1 - y_j) + \sum_{j \in M_{L_i}} x_j \geq 1. \tag{3.16}$$

Proof. The proof for this lemma can be found in the appendix of [53] □

Using Lemma (3.4), we can make the following observations, not found in [53].

Observation 3.5. *If there exists some $j \in \{1,\dots,n\}$ such that $j \in M_{N^+}$ and $j \in M_{L_i}$, then* (3.16) *holds for all x, y.*

Proof. We can rewrite (3.16) as

$$\sum_{j\in M_{N^+}\setminus M_{L_i}} (1-x_i) + \sum_{j\in M_{N^+}} (1-y_j) + \sum_{j\in M_{L_i}\setminus M_{N^+}} x_j + \sum_{j\in M_{N^+}\cap M_{L_i}} ((1-x_j)+x_j) \geq 1. \tag{3.17}$$

In the last component, the x_j's cancel out, meaning the last summation is equal to the cardinality of the intersection between M_{N^+} and M_{L_i}. As the variables x and y have binary bounds, the first 3 summations are all positive and therefore the last summation can act as a lower bound. Hence, if $|M_{N^+} \cap M_{L_i}| \geq 1$, then (3.16) automatically holds for all x, y. □

Observation 3.6. *If there exists some $j \in M_{N^+}$ such that $x_j + y_j \leq 1$, then* (3.16) *holds.*

Proof. (3.16) is equivalent to

$$\sum_{j\in M_{N^+}} (2-x_j-y_j) + \sum_{j\in M_{L_i}} x_j \geq 1. \tag{3.18}$$

If there exists some $j \in M_{N^+}$ such that $x_j + y_j \leq 1$ then $2 - x_j - y_j \geq 1$, implying the above equation holds, thus (3.16) holds. □

Rather than using the relaxation in (3.16), [53] could have used the McCormick constraints to create a linear single-level problem. To do so, let $z_j = x_j y_j$ for all j. (3.15e) is then replaced with

$$\sum_{j=1}^{n} z_j \leq x^\top y^i \quad y^i \in \mathcal{Y}, \tag{3.19a}$$

$$z_j \geq 0 \quad \forall j \in \{1,\dots,n\}, \tag{3.19b}$$

$$z_j \geq x_j + y_j - 1 \quad \forall j \in \{1,\dots,n\}, \tag{3.19c}$$

$$z_j \leq x_j \quad \forall j \in \{1,\dots,n\}, \tag{3.19d}$$

$$z_j \leq y_j \quad \forall j \in \{1,\dots,n\}. \tag{3.19e}$$

We want to show that by using the McCormick constraints, although there will be an additional set of variables, the feasible region is at least as tight as that in [53]. To show this, we show that any point infeasible for (3.16) is also infeasible for (3.19).

Theorem 3.7. *Let* $\mathcal{F}^D = \{(x,y) \in [0,1]^n : \text{(3.15b), (3.15c), (3.16) all hold}\}$ *be the feasible region of* (3.15) *and* $\mathcal{F}^{MC} = \{(x,y) \in [0,1]^n : \text{(3.15b), (3.15c), (3.19) all hold}\}$ *be the feasible region if [53] had used the McCormick constraints, for when the integrality constraints have been dropped. If there exists a solution* $(x,y) \notin \mathcal{F}^D$*, then* $(x,y) \notin \mathcal{F}^{MC}$*.*

Proof. By assumption, let $(x,y) \notin \mathcal{F}^D$. Therefore, (x,y) must violate at least one of (3.15b), (3.15c), (3.16).

If (x,y) violates either (3.15b) or (3.15c) then clearly $(x,y) \notin \mathcal{F}^{Mc}$.

If (x,y) only violates (3.16), then

$$\sum_{j \in M_{N^+}} (1 - x_j) + \sum_{j \in M_{N^+}} (1 - y_j) + \sum_{j \in M_{L_i}} x_j < 1, \tag{3.20}$$

which can be re-arranged to

$$\sum_{j \in M_{N^+}} 1 + \sum_{j \in M_{L_i}} x_j < 1 + \sum_{j \in M_{N^+}} (x_j + y_j - 1). \tag{3.21}$$

By defining $\alpha = |M_{N^+}| - 1$ and using the lower bound $z_j \geq x_j + y_j - 1$ we get

$$\alpha + \sum_{j \in M_{L_i}} x_j < \sum_{j \in M_{N^+}} z_j. \tag{3.22}$$

By definition, $M_{L_i} \subset L_i$ and $\sum_{j=1}^{n} z_j = \sum_{j \in N^+} z_j$. Thus, given that $M_{N^+} \cap M_{L_i} = \emptyset$, using Lemma 3.5 with the fact (x,y) that is infeasible, we get

$$\sum_{j \in L_i} x_j = \sum_{j \in M_{L_i}} x_j + \sum_{j \in L_i \setminus M_{L_i}} x_j, \tag{3.23a}$$

$$\sum_{j \in N^+} z_j = \sum_{j \in M_{N^+}} z_j + \sum_{j \in M_{L_i}} z_j + \sum_{j \in N^+ \setminus \{M_{N^+} \cup M_{L_i}\}} z_j. \tag{3.23b}$$

If the point (x,y) is feasible for the McCormick constraints, then we must have $\sum_{j \in N^+} z_j \leq \sum_{j \in L_i} x_j$. Using this along with (3.23), (3.22) becomes

$$\sum_{j \in L_i \setminus M_{L_i}} x_j - \alpha > \sum_{j \in M_{L_i}} z_j + \sum_{N^+ \setminus \{M_{N^+} \cup M_{L_i}\}} z_j. \tag{3.24}$$

Given that $M_{N^+} \cap M_{L_i} = \emptyset$, then $|M| = |M_{N^+}| + |M_{L_i}|$. As M is a cover of L_i, then $|M| > |L_i|$, which implies $|M_{N^+}| + |M_{L_i}| \geq |L_i| + 1$. As x is binary, then

$$\sum_{j \in L_i \setminus M_{L_i}} x_j \leq |L_i \setminus M_{L_i}| = |L_i| - |M_{L_i}| \leq |M_{N^+}| - 1 = \alpha. \tag{3.25}$$

Thus, $\sum_{j \in L_i \setminus M_{L_i}} x_j - \alpha \leq 0$. However, the right-hand side of (3.24) consists of only positive elements, meaning $\sum_{j \in L_i \setminus M_{L_i}} x_j - \alpha > 0$. This is clearly a contradiction, therefore $\sum_{j \in N^+} z_j \leq \sum_{j \in L_i} x_j$ must not hold and the point (x, y) does not satisfy all of the McCormick constraints and $(x, y) \notin \mathcal{F}^D$. □

Theorem 3.7 shows that any point not in the feasible region $\mathcal{F}^{Mc}$ must also not be in the feasible region $\mathcal{F}^D$. As such, we have that $\mathcal{F}^{Mc} \subseteq \mathcal{F}^D$. However, we can present examples where $\mathcal{F}^{Mc} \subset \mathcal{F}^D$ and there exists some points $(x, y) \in \mathcal{F}^D$ and $(x, y) \notin \mathcal{F}^{Mc}$.

Example 3.4. *Assume we are solving the problem* (3.26) *, i.e. there are no leader or follower constraints and* $\mathcal{Y} = \{y'\}$ *where* $y' = (0, 1, 1, \ldots, 1)$*. By Lemma 3.5, if the cover* M *is such that there exists some* $j \in M_{N^+}$ *and* $j \in M_{L_i}$ *then all* (x, y) *satisfy* (3.16)*. Thus, construct such an* M *and observe the point* (x, y) *where* $x = y = (1, \ldots, 1)$*. By Lemma 3.5* $(x, y) \in \mathcal{F}^D$*. However, using the McCormick constraints we get* $z = (1, \ldots, 1)$*, which causes the value function constraint to be* $n \leq n - 1$*. Thus, the point* $(x, y) \notin \mathcal{F}^{Mc}$*.*

$$\max_x \quad d_1^\top x + d_2^\top y \tag{3.26a}$$

$$s.t. \quad x, y \in \{0, 1\}^n \tag{3.26b}$$

$$x^\top y \leq x^\top y^i \quad \forall y^i \in \mathcal{Y}. \tag{3.26c}$$

3.7.2 McCormick over Summations

Lets assume that there exists some variables, x and y, where $x \in [0, M]^n$ and $y \in \{0, 1\}^{m \times n}$ and we have the product $w = \sum_{j=1}^n \sum_{i=1}^m y_{ij} x_j$. As we can see, x does not depend on i, so we can write w as either $\sum_{j=1}^n \sum_{i=1}^m (y_{ij} x_j)$ or $\sum_{j=1}^n x_j (\sum_{i=1}^m y_{ij})$, where in the first case we take the product between x and y then sum over j and i and in the latter we take the product between x and the summation of y over i before taking the sum over j. Therefore, if we wish to linearise this expression using the McCormick envelopes, we can either 1) apply them directly using the product between $y_j^{ik} x_j$ or

2) perform a binary expansion on $\sum_{i=1}^{m} y_{ij}$ and then apply the McCormick envelopes between x and the generated binary variables.

Applying the McCormick constraints directly, we get

$$z_{ij} \geq 0 \quad \forall i \in \{1,\dots,m\}, j \in \{1,\dots,n\}, \tag{3.27a}$$
$$z_{ij} \geq x_j - M(1-y_{ij}) \quad \forall i \in \{1,\dots,m\}, j \in \{1,\dots,n\}, \tag{3.27b}$$
$$z_{ij} \leq x_j \quad \forall i \in \{1,\dots,m\}, j \in \{1,\dots,n\}, \tag{3.27c}$$
$$z_{ij} \leq My_{ij} \quad \forall i \in \{1,\dots,m\}, j \in \{1,\dots,n\}, \tag{3.27d}$$
$$w = \sum_{i=1}^{m}\sum_{j=1}^{n} z_{ij}, \tag{3.27e}$$

whereas performing a binary expansion and then the McCormick envelopes, we get

$$\sum_{i=1}^{m} y_{ij} = \sum_{l=1}^{\lfloor \log_2 m \rfloor + 1} 2^{l-1}\alpha_{lj} \quad \forall j \in \{1,\dots,n\}, \tag{3.28a}$$
$$z_{lj} \geq 0 \quad \forall l \in \{1,\dots,\lfloor \log_2 m \rfloor + 1\}, j \in \{1,\dots,n\}, \tag{3.28b}$$
$$z_{lj} \geq x_j - M(1-\alpha_{lj}) \quad \forall l \in \{1,\dots,\lfloor \log_2 m \rfloor + 1\}, j \in \{1,\dots,n\}, \tag{3.28c}$$
$$z_{lj} \leq x_j \quad \forall l \in \{1,\dots,\lfloor \log_2 m \rfloor + 1\}, j \in \{1,\dots,n\}, \tag{3.28d}$$
$$z_{lj} \leq M\alpha_{lj} \quad \forall l \in \{1,\dots,\lfloor \log_2 m \rfloor + 1\}, j \in \{1,\dots,n\}, \tag{3.28e}$$
$$w = \sum_{j=1}^{n}\sum_{l=1}^{\lfloor \log_2 m \rfloor + 1} 2^{l-1} z_{lj}. \tag{3.28f}$$

From here on in, we shall refer to (3.27) as Direct McCormicks, DMC, and (3.28) as Summation McCormicks, SMC. In the DMC, we have introduced mn many additional variables, whereas using the SMC method, we have introduced $2(\lfloor \log_2 m \rfloor + 1)n$ many variables, with $(\lfloor \log_2 m \rfloor + 1)n$ many coming from α and the other $(\lfloor \log_2 m \rfloor + 1)n$ coming from z. Likewise, with the DMC method we would be introducing an additional $4mn$ constraints, whereas in the SMC we would be introducing $4(\lfloor \log_2 m \rfloor + 1)n + n$ constraints.

Now, we want to compare the polyhedra created by both of these methods and see if we can deduce, which is tighter. To do this, we shall perform Fourier–Motzkin elimination, leaving us with an upper and lower bound for w, for both methods, which we can compare. Substituting (3.27a)–(3.27d) into (3.27e), we get the bounds for the DMC to be

$$w \geq 0, \tag{3.29a}$$

$$w \geq \sum_{j=1}^{n} \sum_{i=1}^{m} (x_j - M(1 - y_{ij})), \tag{3.29b}$$

$$w \leq \sum_{j=1}^{n} \sum_{i=1}^{m} x_j, \tag{3.29c}$$

$$w \leq \sum_{j=1}^{n} \sum_{i=1}^{m} M y_{ij}, \tag{3.29d}$$

and substituting (3.28b)–(3.28e) into (3.28f), we get the bounds for SMC to be

$$w \geq 0, \tag{3.30a}$$

$$w \geq \sum_{j=1}^{n} \sum_{l=1}^{\lfloor \log_2 m \rfloor + 1} 2^{l-1} (x_j - M(1 - \alpha_{lj})), \tag{3.30b}$$

$$w \leq \sum_{j=1}^{n} \sum_{l=1}^{\lfloor \log_2 m \rfloor + 1} 2^{l-1} x_j, \tag{3.30c}$$

$$w \leq \sum_{j=1}^{n} \sum_{l=1}^{\lfloor \log_2 m \rfloor + 1} 2^{l-1} M \alpha_{ij}. \tag{3.30d}$$

In both sets of bounds, we have $w \geq 0$, so this can be ignored. Likewise, if we substitute (3.28a) into (3.30d), we get $w \leq \sum_{j=1}^{n} \sum_{i=1}^{m} M y_{ij}$, which is the same as (3.29d) and thus, can also be ignored. This leaves us to compare the lower bounds (3.29b) and (3.30b) plus the upper bounds (3.29c) and (3.30c).

Focusing on the upper bounds, (3.29c) can be written as $m \sum_{j=1}^{n} x_j$, as x is independent of i and (3.30c) can be written as $(\sum_{l=1}^{\lfloor \log_2 m \rfloor + 1} 2^{l-1}) \sum_{j=1}^{n} x_j$ for the same reasons, with respect to l. Now we can see that the tightness is dependent on which of m and $\sum_{l=1}^{\lfloor \log_2 m \rfloor + 1} 2^{l-1}$ is smaller. Given that we achieved the latter as a result of the binary expansion of y, we can see that, if $m = 2^k - 1$, for some $k \in \mathbb{N}$, then $m = \sum_{l=1}^{\lfloor \log_2 m \rfloor + 1} 2^{l-1}$ and both DMC and SMC provide the same upper bounds on w. Otherwise, we must have $m < \sum_{l=1}^{\lfloor \log_2 m \rfloor + 1} 2^{l-1}$ and thus the DMC shall provide a tighter upper bound.

For the lower bounds, in (3.30b) we can substitute in (3.28a), to get $w \geq \sum_{j=1}^{n} \sum_{l=1}^{\lfloor \log_2 m \rfloor + 1} 2^{l-1}(x_j - M) + M \sum_{i=1}^{m} y_{ij}$, which means, after cancelling out the last term, we are comparing $m \sum_{j=1}^{n} (x_j - M)$ and $(\sum_{l=1}^{\lfloor \log_2 m \rfloor + 1} 2^{l-1}) \sum_{j=1}^{n} (x_j - M)$ for the lower bounds of DMC and SMC respectively. Once again, we get the terms m and $\sum_{l=1}^{\lfloor \log_2 m \rfloor + 1} 2^{l-1}$, however, because we are now observing the tightness with respect to the lower bound, we want to use which of these is larger. As before, if $m = 2^k - 1$, for

some $k \in \mathbb{N}$ then they both achieve the same tightness, otherwise $m < \sum_{l=1}^{\lfloor \log_2 m \rfloor + 1} 2^{l-1}$ and the SMC shall provide a tighter lower bound.

Now, if we look at the absolute gap between the upper and lower bounds, we find that for DMC the difference between the upper and lower bound is $nMm - M\sum_{j=1}^{n}\sum_{i=1}^{m} y_{ij}$ and $nM\sum_{l=1}^{\lfloor \log_2 m \rfloor + 1} 2^{l-1} - M\sum_{j=1}^{n}\sum_{i=1}^{m} y_{ij}$ for SMC. Therefore, if $m = 2^k - 1$, for some $k \in \mathbb{N}$, the gap between the upper and lower bounds are the same in both DMC and SMC, otherwise, DMC has a tighter gap.

In conclusion, if $m = 2^k - 1$, for some $k \in \mathbb{N}$, then both DMC and SMC provide the same bounds, otherwise, if we are maximising w, we want a tighter upper bound and thus would prefer DMC, whereas if we are minimising we would want a tighter lower bound and would prefer SMC.

3.7.2.1 y Constraint

Should there exist some constraint $\sum_{i=1}^{m} y_{ij} \leq s$ for all j, then when we perform the binary expansion on y, we would perform the summation between $l = 1$ and $l = \lfloor \log_2 s \rfloor + 1$ rather than m. Therefore, should such a constraint exist, we can use the same argument to deduce that, if $s \leq 2^{\lfloor \log_2 (m+1) \rfloor} - 1$, then by the same arguments we have just presented, SMC shall provide an upper bound at least as strong as DMC and DMC shall provide a lower bound at least as strong as SMC.

In the following chapter, we present solution methods for solving the unit and non-unit supply cases. A majority of these solution methods focus around using the HPR, which shall be solved using sophisticated techniques. As such, the problem which is to be solved, the HPR, is a maximisation problem. Therefore, when $s < 2^{\lfloor \log_2 (m+1) \rfloor} - 1$, we would predict the SMC formulation to reach a solution quicker than the DMC and visa versa when $s > 2^{\lfloor \log_2 (m+1) \rfloor} - 1$. At equality, they should both provide the same upper bound and should have similar computational times.

3.7.2.2 McCormick Linearisations

Thus far we have developed a single formulation, (3.3), for the unit supply case and 2 formulations, (3.5) and (3.9) for the non-unit supply case. For (3.3) and (3.5), we have the non-linear terms $\sum_{j\in\mathcal{J}}\sum_{i\in\mathcal{I}}\sum_{k=1}^{d^i} y_j^{ik} x_j$ and $\sum_{j\in\mathcal{J}}\sum_{i\in\mathcal{I}}\sum_{k=1}^{d^i} \bar{y}_j^{ik} x_j$ respectively. In both of these cases, we have a product between 2 variables, over 3 summations when x only depends on j. Therefore we can apply both the DMC and SMC linearisation methods to compare their performance. As such, (A.1) and (A.2) represent the unit supply reformulations using DMC and SMC, respectively, and (A.5) and (A.6) represent the DMC and SMC reformulations of (3.5), respectively. For formulation (3.9) we have no need for any SMC's and thus only use DMC's in (A.7).

Chapter 4

Solution Methods

In this chapter, we present various solution methods for solving the bilevel pricing problems presented in Chapter 3. To begin with, we shall present solution methods for both the unit and non-unit supply cases that primarily use cutting planes. These methods begin by solving the HPR within a Branch-and-Cut framework, where value function cutting planes are introduced with the discovery of all bilevel infeasible solutions. For the unit supply case, this translates as a simple cutting plane algorithm as the feasible regions of the followers are independent of the leader's variables and thus unperturbed. However, for the non-unit supply case these cutting planes are not globally valid, and therefore we must introduce additional variables to indicate when these constraints should be active. As such, we introduce two variable generation methods, where the cutting planes either focus on the exact response of the followers or rather the set of commodities. Following this, we also demonstrate how we can achieve the same results by using sophisticated branching strategies that remove the need for additional variables.

After this, we present solution methods focused around the leader having pre-computed a subset of follower solutions. These methods break the bilevel problem into two sub-problems. In the first one, we assume that the followers respond with a solution contained with the pre-computed subset. As the followers possible responses have been pre-computed, we can show that the resulting sub-problem admits a totally unimodular follower's problem, allowing for a wider range of reformulation methods, other than just the value function approach. For the second sub-problem, we assume the followers react with solutions not in the subset. This can be solved using one of the variable generation or branching solution methods just described.

Finally, we demonstrate how we can contain the "pre-computed" solution method within a single problem using another branching strategy.

In Chapter 5 we present computational results comparing the performance of the formulations presented in the previous chapter and the proceeding solution methods.

All of these solution methods stem from solving the HPR and using cutting planes or branching strategies to reach the bilevel solution. As such, we use the state-of-the-art solver CPLEX to solve the HPR, given its use of callback functions [89].

Cutting planes are introduced using the *LazyConstraintCallback*, which CPLEX calls at every integer point. Within this callback, we can introduce a linear constraint globally using the *add* function, which shall be used for constraint (4.1) in Algorithm CP.

For the n-ary, Improved n-ary and KSB methods, we use a sophisticated branching strategy which is performed using *BranchCallback*. This callback is called at every node CPLEX wants to branch on. At nodes we wish to deviate from CPLEX's branching strategy, we use the *make_branch* function, which allows the user to create two branches with user-defined local constraints for each child node. Here, we introduce the constraints (4.11) for the n-ary and Improved n-ary methods and the constraint given in Figure 4.3 for the KSB method.

When introducing these branches, we need to make sure that we are at the correct node, as we do not alter CPLEX's node selection. To do this, we use the *get_node_data* and *set_node_data* functions, which allow us to set, and read, a label to a node, which can be used to ensure the correct branches are being created.

For nodes where we do not wish to invoke any specific branching strategy and would like CPLEX to carry on with their usual procedures, i.e. fractional nodes, we can use the *make_cplex_branch* function.

We also use the *IncumbentCallback* to ignore any integer feasible solution which CPLEX may appear during a branching process. This can occur when we are implementing a n-ary branching strategy and CPLEX discovers an integer solution before we have generated all n branches. In such instances, we want to ignore the incumbent, which can be done using the *reject* function, and then continue to generate the remaining branches.

4.1 Unit Supply

For the unit supply case, as we have already discussed, the follower's feasible region is independent of the leader's variables. As a result, we can solve the bilevel problem by solving the value function reformulation given by (2.21). Note, as a result of the independent feasible region, constraints (2.21b) and (2.21b) are actually $G_i^U(x) \leq 0$ for all i and $G_j^L(y) \leq 0$ for all j, respectively. If y is continuous, our problem admits similarities with Semi-Infinite Programming, SIP, and the proceeding cutting plane algorithm would is similar the general algorithm found in [23]. However, generating all of the feasible solutions for the followers is clearly computationally expensive. Thus, we propose an approach, where we introduce cutting planes on the fly until we reach a bilevel

feasible solution. Although this solution method can be used with either formulations (A.1) or (A.2), where the non-linear components of the cutting plane can be linearised in the necessary fashion, we shall refer to the non-linear formulation (3.3) for ease of reading.

We begin by solving the HPR of (3.3) using a standard Branch-and-Cut approach. At any integer solution $(x^*, \bar{x}^*, y^*)$ discovered, we check for bilevel feasibility by solving the corresponding follower's problem. If we discover that for $(x^*, \bar{x}^*, y^*)$ there exists a better response $\tilde{y}$ for the follower, we need to introduce a cutting plane of the form $F^L(x, y) \leq F^L(x, \tilde{y})$. Should $(x^*, \bar{x}^*, y^*)$ be bilevel feasible then we update the incumbent and bounds if necessary and continue through the tree.

Algorithm CP (Cutting plane)

1. Create the HPR of (3.3) and begin solving using a Branch-and-Cut framework.
2. Let $(x^*, \bar{x}^*, y^*)$ be the solution at any given node. If $(x^*, \bar{x}^*, y^*)$ is fractional, then carry on with the Branch-and-Cut process. Else, solve the follower's problem $\min_y\{(3.3f) : (3.3g) - (3.3i)\}$ for fixed $(x, \bar{x}) = (x^*, \bar{x}^*)$ and let $\tilde{y}$ be the solution.
3. If $\sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} \sum_{j\in\mathcal{J}} y_j^{*\,ik}(x_j^* + c_j) \leq \sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} \sum_{j\in\mathcal{J}} \tilde{y}_j^{ik}(x_j^* + c_j)$ then the solution $(x^*, \bar{x}^*, y^*)$ is bilevel feasible, the upper/lower bounds, along with the incumbent, can be updated if necessary and we continue with the Branch-and-Cut process.
4. Else, $(x^*, \bar{x}^*, y^*)$ is bilevel infeasible and we remove it from the feasible region by introducing, globally, the cutting plane

$$\sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} \sum_{j\in\mathcal{J}} y_j^{ik}(x_j + c_j) \leq \sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} \sum_{j\in\mathcal{J}} \tilde{y}_j^{ik}(x_j + c_j), \tag{4.1}$$

where $\tilde{y} \in \psi(x^*, \bar{x}^*)$. Carry on with the Branch-and-Cut approach and return to Step 2 when necessary.

Proposition 4.1. *The CP algorithm will terminate at an optimal bilevel solution of* (3.3) *in a finite number of steps.*

Proof. As discussed, the HPR of (3.3) is just a relaxation of single level value function reformulation. If the optimal solution to the HPR is bilevel feasible then it is also the optimal solution to the original bilevel problem. In the worst case scenario, we would iterate through CP until we have the cutting plane (4.1) for every feasible follower solution. At which point, the HPR is now equivalent to the value function reformulation and the optimal solution will also be the optimal solution to the bilevel problem.

In terms of the number of iterations that may be needed, as we have assumed that the feasible region for the follower is bounded and discrete, as the variables are binary,

then this implies that there are a finite number of solutions. Therefore, there will always be a finite number of maximum cutting planes that will be needed to reach the value function reformulation. □

4.2 Non-Unit Supply

In Section 4.1, the cutting plane (4.1) introduced within the Branch-and-Cut framework restricted the value of the follower's objective function to that of a feasible follower solution that has been found. In the context of the unit supply case, the feasible regions of the followers are independent of the leaders choice, thus any follower solution discovered is always a feasible response for the follower and the corresponding cutting plane, for each discovered solutions will always be globally valid.

However, once we have introduced duplicate commodities, we have to distinguish between commodities that the follower purchases from the leader, and from the market, which clearly depends on the amount that the leader has bought. This leads us to using the max operator in (3.4). If we want to use the same cutting plane method, using (4.1), we would need to embed the max operator on both the LHS and RHS to give us

$$\begin{aligned} &\sum_{j\in\mathcal{J}}\Big(\max\{0,\sum_{i\in\mathcal{I}}\sum_{k=1}^{d^i}y_j^{ik}-(s_j-\bar{x}_j)\}x_j+\sum_{i\in\mathcal{I}}\sum_{k=1}^{d^i}y_j^{ik}c_j\Big)\\ \leq &\sum_{j\in\mathcal{J}}\Big(\max\{0,\sum_{i\in\mathcal{I}}\sum_{k=1}^{d^i}{y'}_j^{ik}-(s_j-\bar{x}_j)\}x_j+\sum_{i\in\mathcal{I}}\sum_{k=1}^{d^i}{y'}_j^{ik}c_j\Big)\quad\forall y'\in\psi(x^*,\bar{x}^*). \end{aligned} \tag{4.2}$$

In the previous chapter, we presented two formulations to handle the max operator on the left-hand side. However, the max operator on the right-hand side is new, as the only variable is $\bar{x}$, as y' is fixed. As such, we cannot directly use this constraint within the cutting plane procedure described previously. Thus, we present four further solution methods which shall use either a column generation approach or a sophisticated branching strategy.

Note, the following solution methods can both be used for the reformulations given by (3.5) and (3.9), however, we shall focus our attention to the dichotomic formulation.

4.2.1 Non-Symmetry-Free Cutting Plane

Assume that the HPR of (3.5) is being solved within a Branch-and-Cut framework and we discover an integer solution $(x^*,\bar{x}^*,\bar{y}^*,y^*)$. In solving the follower's problem, parameterised by $(x^*,\bar{x}^*)$, we discover that $(\tilde{\bar{y}},\tilde{y})$ returns a smaller follower's objective.

As a result of the follower's feasible region no longer being independent of the leaders variables, the equivalent value function constraint we presented earlier is no longer globally valid. Thus, we can only apply this constraint to parts of the feasible region that allow the follower to reply with the solution $(\tilde{y}, \tilde{y})$, i.e.,

$$\begin{aligned}\sum_{j\in\mathcal{J}}\Big(\sum_{i\in\mathcal{I}}\sum_{k=1}^{d^i}\bar{y}_j^{ik}(x_j+c_j)+\sum_{i\in\mathcal{I}}\sum_{k=1}^{d^i}y_j^{ik}c_j\Big) &\leq \sum_{j\in\mathcal{J}}\Big(\sum_{i\in\mathcal{I}}\sum_{k=1}^{d^i}\tilde{y}_j^{ik}(x_j+c_j)+\sum_{i\in\mathcal{I}}\sum_{k=1}^{d^i}\tilde{y}_j^{ik}c_j\Big)\\ \iff \quad \forall j\in\mathcal{J} \quad \sum_{i\in\mathcal{I}}\sum_{k=1}^{d^i}\tilde{y}_j^{ik} &\leq \bar{x}_j \wedge \sum_{i\in\mathcal{I}}\sum_{k=1}^{d^i}\tilde{y}_j^{ik} \leq s_j-\bar{x}_j.\end{aligned} \tag{4.3}$$

On the left-hand side, we have the value function constraint which states that the cost of the followers response $(\bar{y}, y)$ must be less than or equal to the cost of the solution $(\tilde{y}, \tilde{y})$. However, this constraint should only be active for the parts of the feasible region where $(\tilde{y}, \tilde{y})$ is a feasible follower response. Remembering that $\tilde{y}$ represents the number of commodities that the follower purchases from the leader, then clearly $(\tilde{y}, \tilde{y})$ is only feasible when the leader owns more commodities than $\tilde{y}$, i.e. $\sum_{i\in\mathcal{I}}\sum_{k=1}^{d^i}\tilde{y}_j^{ik} \leq \bar{x}_j$ for all $j \in \mathcal{J}$. Likewise, we can employ the same argument for the commodities the follower does not purchase from the leader, giving us $\sum_{i\in\mathcal{I}}\sum_{k=1}^{d^i}\tilde{y}_j^{ik} \leq s_j - \bar{x}_j$ for all $j \in \mathcal{J}$.

For the Non-Symmetry Free Cutting Plane, NSFCP, method, we shall enforce this constraint by introducing three sets of binary variables, δ, $\delta^{\tilde{y}}$ and $\delta^{\tilde{\bar{y}}}$, which shall be used as triggers for when the first part of (4.3) should be active, where

$$\delta_j^{\tilde{y}} = 1 \quad \Longrightarrow \sum_{i\in\mathcal{I}}\sum_{k=1}^{d^i}\tilde{y}_j^{ik} \leq \bar{x}_j, \tag{4.4a}$$

$$\delta_j^{\tilde{y}} = 0 \quad \Longrightarrow \sum_{i\in\mathcal{I}}\sum_{k=1}^{d^i}\tilde{y}_j^{ik} \geq \bar{x}_j + 1, \tag{4.4b}$$

$$\delta_j^{\tilde{\bar{y}}} = 1 \quad \Longrightarrow \sum_{i\in\mathcal{I}}\sum_{k=1}^{d^i}\tilde{\bar{y}}_j^{ik} \leq s_j - \bar{x}_j, \tag{4.4c}$$

$$\delta_j^{\tilde{\bar{y}}} = 0 \quad \Longrightarrow \sum_{i\in\mathcal{I}}\sum_{k=1}^{d^i}\tilde{\bar{y}}_j^{ik} \geq s_j - \bar{x}_j + 1, \tag{4.4d}$$

$$\delta_j = \delta_j^{\tilde{\bar{y}}}\delta_j^{\tilde{y}}, \tag{4.4e}$$

for all $j \in \mathcal{J}$. This can be done using the linear constraints

$$\bar{M}\delta_j^{\tilde{\bar{y}}} \geq \bar{x}_j + 0.5 - \sum_{i\in\mathcal{I}}\sum_{k=1}^{d^i} \tilde{\bar{y}}_j^{ik}, \tag{4.5a}$$

$$\bar{M}(1-\delta_j^{\tilde{\bar{y}}}) \geq -\Big(\bar{x}_j + 0.5 - \sum_{i\in\mathcal{I}}\sum_{k=1}^{d^i} \tilde{\bar{y}}_j^{ik}\Big), \tag{4.5b}$$

$$\bar{M}\delta_j^{\tilde{y}} \geq s_j - \bar{x}_j + 0.5 - \sum_{i\in\mathcal{I}}\sum_{k=1}^{d^i} \tilde{y}_j^{ik}, \tag{4.5c}$$

$$\bar{M}(1-\delta_j^{\tilde{y}}) \geq -\Big(s_j - \bar{x}_j + 0.5 - \sum_{i\in\mathcal{I}}\sum_{k=1}^{d^i} \tilde{y}_j^{ik}\Big), \tag{4.5d}$$

$$\mathcal{MC}(\delta_j, \delta_j^{\tilde{y}}, \delta_j^{\tilde{\bar{y}}}), \tag{4.5e}$$

for all $j \in \mathcal{J}$. The value of $\bar{M}$ must be such that $\bar{M}$ is always greater than the right hand side of both (4.5a) and (4.5c), which can be done by defining $\bar{M} = \max\{s_j : j \in 1,\ldots,n\} + 1$. Thus, the cutting plane to be introduced is

$$\begin{aligned}\sum_{j\in\mathcal{J}}\Big(\sum_{i\in\mathcal{I}}\sum_{k=1}^{d^i}\bar{y}_j^{ik}(x_j+c_j) + \sum_{i\in\mathcal{I}}\sum_{k=1}^{d^i} y_j^{ik}c_j\Big) \leq \sum_{j\in\mathcal{J}}\Big(\sum_{i\in\mathcal{I}}\sum_{k=1}^{d^i}\tilde{\bar{y}}_j^{ik}(x_j+c_j) + \sum_{i\in\mathcal{I}}\sum_{k=1}^{d^i}\tilde{\bar{y}}_j^{ik}c_j\Big) \\ + M^{\delta}\Big(\sum_{j\in\mathcal{J}}(1-\delta_j)\Big)\end{aligned} \tag{4.6}$$

where M^{δ} is a constant that is large enough such that if $\sum_{j\in\mathcal{J}}(1-\delta_j) \geq 1$, the right-hand side of the cutting plane is sufficiently large that all solutions are feasible and this constraint is effectively inactive. Calculating a value for M^{δ} can be found by setting $M^{\delta} > M^{\delta+} - M^{\delta-}$ where

$$M^{\delta+} = \max_{x,\tilde{x}} \quad \sum_{j\in\mathcal{J}}\Big(\sum_{i\in I}\sum_{k=1}^{d^i}\bar{y}_j^{ik}(x_j+c_j) + y_j^{ik}c_j\Big) \tag{4.7a}$$

$$\text{s.t.} \quad (3.4b) - (3.4f) \tag{4.7b}$$

$$(3.5d) - (3.5i) \tag{4.7c}$$

$$\tag{4.7d}$$

and

$$M^{\delta} = \min_{y} \quad \sum_{j \in \mathcal{J}} \left(\sum_{i \in I} \sum_{k=1}^{d^i} y_j^{ik} c_j \right) \tag{4.8a}$$

$$\text{s.t.} \quad \text{(3.5d)} - \text{(3.5i)}. \tag{4.8b}$$

(4.8) is very similar to the HPR, however its objective is the followers rather than the leaders, with $M^{\delta+}$ equalling the maximum possible objective for the follower. Whereas $M^{\delta-}$ is the minimum possible objective for the follower, meaning that $M^{\delta+} - M^{\delta-}$ is a sufficient lower bound for M^{δ}.

If $\sum_{j \in \mathcal{J}}(1 - \delta_j) = 0$, i.e. $\bar{x}$ allows for $(\tilde{y}, \tilde{y})$ to be a feasible solution for the follower, we have constrained the followers objective to be less than that achieved with $(\tilde{y}, \tilde{y})$. Thus, the NSFCP solution method can be described as follows.

Algorithm NSFCP (Non-Symmetry Free Cutting Plane)

1. Create the HPR of (3.5) and solve using a Branch-and-Cut framework.
2. Let $(x^*, \bar{x}^*, \bar{y}^*, y^*)$ be the solution at any given node. If $(x^*, \bar{x}^*, \bar{y}^*, y^*)$ is fractional, then carry on with the Branch-and-Cut process. Else, solve the follower's problem $\min_y\{\text{(3.5c)} : \text{(3.5d)} - \text{(3.5i)}\}$ and let $\phi(x^*, \bar{x}^*)$ be an optimal objective value and $(\tilde{y}, \tilde{y})$ be the optimal response.
3. If $\sum_{j \in \mathcal{J}} \sum_{i \in \mathcal{I}} \sum_{k=1}^{d^i} \left(\bar{y}^*{}_j^{ik}(x_j^* + c_j) + y^*{}_j^{ik} c_j \right) \leq \phi(x^*, \bar{x}^*)$, then the solution $(x^*, \bar{x}^*, \bar{y}^*, y^*)$ is bilevel feasible, the upper/lower bounds, along with the incumbent can be updated if necessary and we continue with the Branch-and-Cut process.
4. Else, $(x^*, \bar{x}^*, \bar{y}^*, y^*)$ is bilevel infeasible and we stop the Branch-and-Cut framework. We introduce the variables $\delta, \delta^{\tilde{y}} \delta^{\tilde{y}}$, along with the constraints (4.5) and (4.6), restart the Branch-and-Cut framework with the new variables and constraints and return to Step 2 when necessary.

This approach is similar to that of [105] who also use a relaxed formulation of their original bilevel problem, introducing cutting planes on the fly along with additional variables to indicate whether or not such cutting planes should be active. In [44] we can find an application to a trilevel problem related to airlines, which uses binary variables to force constraints to have slack.

With this solution method, we solve the follower's problem with every integer solution discovered in the Branch-and-Cut tree used to solve the HPR. Should this solution be bilevel infeasible, then we halt the solving of this HPR and generate a new one with the necessary value function constraint and variables. In other solution methods, the

followers problem is only solved once the HPR has been solved to optimality [105, 159]. In not solving the HPR to optimality, we hope that the algorithm shall reach an HPR with a sufficient number of value function constraint such that we can obtain the bilevel optimal solution quicker.

Using our variable generation approach, for every bilevel infeasible solution, we introduce $3n$ binary variables along with $8n + 1$ constraints, $8n$ from (4.5) and the single value function constraint (4.6). Once again, similar to the CP algorithm, assuming the follower's region is bounded, then there exists a finite number of follower solutions and thus the algorithm shall terminate in a finite number of steps.

4.2.2 Symmetry-Free Cutting Plane

With the NSFCP method, the right-hand side of (4.6) is fixed to the cost of the exact response that the follower selected for the leader's given variables $(x, \bar{x})$ and the additional $\delta, \delta^{\bar{y}}\delta^{\tilde{y}}$ variables are used to indicate when this cutting plane shall be active. This means that we could potentially have a situation where we have multiple cutting planes that consist of the same subset of commodities, but because a different amount was purchased from the leader in these solutions, we have had to treat them separately.

For example, assume there exists an instance that for a given set of leader's variables, when checking bilevel feasibility the follower purchases commodity j_1 from the leader and commodity j_2 from the market. We then perform the column generation and introduce the cutting plane (4.6). Now, at another bilevel feasibility check, the follower responds by purchasing commodity j_1 from the market and commodity j_2 from the leader. In reality, the follower has purchased the same two commodities in both responses. However, as they have bought differing amount from the leader, using the above method, we would have to treat these responses separately and thus introduce two cutting planes, along with two sets of variables.

Therefore, the aim of the Symmetry-Free Cutting Plane, SFCP, method is to focus on the set of commodities in the solution $(\tilde{y}, \tilde{y})$, rather than the exact response $(\tilde{y}, \tilde{y})$. Note, although the follower's feasible region is technically dependent on the leaders variables, any set of commodities that provides a feasible solution for the follower's problem shall always be a feasible set. Thus, for a given follower solution $(\tilde{y}, \tilde{y})$ we introduce the integer variables $(\bar{v}, v)$ which represent how the follower's should react if restricted to the commodities in $(\tilde{y}, \tilde{y})$. Here, $\bar{v}$ represents the commodities that would have been bought from the leader and v from the market. This can be enforced using the constraints

$$\bar{v}_j + v_j = \sum_{i \in \mathcal{I}} \sum_{k=1}^{d^i} (\tilde{y}_j^{ik} + \tilde{y}_j^{ik}) \quad \forall j \in \mathcal{J}, \tag{4.9a}$$

$$(s_j - \bar{x}_j) - v_j \leq s_j(1 - \delta_j) \quad \forall j \in \mathcal{J}, \tag{4.9b}$$

$$\bar{v}_j \leq \delta_j \bar{x}_j \quad \forall j \in \mathcal{J}, \tag{4.9c}$$

$$v_j \leq s_j - \bar{x}_j \quad \forall j \in \mathcal{J}. \tag{4.9d}$$

Constraint (4.9a) ensures that the same commodities which appear in the solution $(\tilde{y}, \tilde{y})$ shall also appear in $(\bar{v}, v)$. As before, the follower shall only purchase commodities from the leader if the amount they require is greater than what the leader has not bought. Hence $\bar{v}_j > 0 \iff v_j = s_j - \bar{x}_j$. Therefore, to know when this happens we have introduced another variable δ along with constraint (4.9b). When $v_j < s_j - \bar{x}_j$, i.e. the follower would not need to purchase commodity j from the leader, the left-hand side of (4.9b) is greater than 0, forcing $\delta_j = 0$. However, when $v_j = s_j - \bar{x}$, δ_j's value can be either 0 or 1, but because we are maximising the leaders objective, δ_j shall be equal to 1 as this gives the leader an objective at least as large as when $\delta_j = 0$. Constraints (4.9c) and (4.9d) ensure that the follower would not purchase more from the leader/market than what the leader/market have, respectively. The resulting cutting plane to be used is given by

$$\sum_{j \in \mathcal{J}} \Big(\sum_{i \in \mathcal{I}} \sum_{k=1}^{d^i} \bar{y}_j^{ik}(x_j + c_j) + \sum_{i \in \mathcal{I}} \sum_{k=1}^{d^i} y_j^{ik} c_j \Big) \leq \sum_{j \in \mathcal{J}} \Big(\bar{v}_j(x_j + c_j) + v_j c_j \Big). \tag{4.10}$$

On the left-hand side, we have the followers objective, given the leaders taxation x. Whereas on the right-hand side, we have the cost of the optimal response for the follower if they were restricted to the commodities in $(\tilde{y}, \tilde{y})$. In contrast to the cutting plane given by (4.3), this is globally valid and does not require any additional Big-M components. Hence, the SFCP solution method can be described as follows.

Algorithm SFCP

1. Create the HPR of (3.5) and solve using a Branch-and-Cut framework.
2. Let $(x^*, \bar{x}^*, \bar{y}^*, y^*)$ be the solution at any given node. If $(x^*, \bar{x}^*, \bar{y}^*, y^*)$ is fractional, then carry on with the Branch-and-Cut process. Else, solve the followers problem $\min_y\{(3.5c) : (3.5d) - (3.5i)\}$ and let $\phi(x^*, \bar{x}^*)$ be the optimal objective value and $(\tilde{y}, \tilde{y})$ be the optimal response.
3. If $\sum_{j\in\mathcal{J}}\sum_{i\in\mathcal{I}}\sum_{k=1}^{d^i}\left(\bar{y}^{*\,ik}_j(x^*_j + c_j) + y^{*\,ik}_j c_j\right) \leq \phi(x^*, \bar{x}^*)$, then the solution $(x^*, \bar{x}^*, \bar{y}^*, y^*)$ is bilevel feasible, the upper/lower bounds, along with the incumbent can be updated if necessary and we continue with the Branch-and-Cut process.
4. Else, $(x^*, \bar{x}^*, \bar{y}^*, y^*)$ is bilevel infeasible and we stop the Branch-and-Cut framework. We introduce the variables $\bar{v}_j, v_j, \delta$, along with the constraints (4.9) and (4.10), restart the Branch-and-Cut framework with the new variables and constraints and return to Step 2 when necessary.

With every bilevel infeasible solution found, we are introducing $n(1 + 2\sum_{i\in\mathcal{I}} d^i)$ many variables along with $4n + 1$ many constraints. Therefore $i = 1$ and $d^1 = 1$ is the only instance where the SFCP method shall introduce fewer variables with every cutting plane. Whereas, the SFCP shall introduce $4n$ fewer constraints with every cutting plane compared to the NSFCP method.

4.2.3 n-ary Branching

With the NSFCP and SFCP methods, to enforce the relevant value function constraints, we have had to introduce a series of variables and constraints. As such, we have to repeatedly generate and solve MILPs until we reach the bilevel optimal solution. However, rather than introducing any additional variables, we can invoke a sophisticated branching strategy, which can solve our problem within a single Branch-and-Cut tree, where we branch on the $\delta^{\bar{y}}$ and $\delta^{\tilde{y}}$ variables from the NSFCP method, without adding the variables. This method is similar to that found in [43], where they present a Branch-and-Bound algorithm.

Let us assume we are solving the HPR of (3.5) and at some node S we discover an integer solution $(x^*, \bar{x}^*, \bar{y}^*, y^*)$ which is bilevel infeasible as the follower has a better response in the form of $(\tilde{y}, \tilde{y})$. As shown in (4.3), we can only apply the value function constraint if $\sum_{i\in\mathcal{I}}\sum_{k=1}^{d^i}\tilde{y}^{ik}_j \leq \bar{x}_j$ and $\sum_{i\in\mathcal{I}}\sum_{k=1}^{d^i}\tilde{y}^{ik}_j \leq (s_j - \bar{x}_j)$ are satisfied for all $j \in \mathcal{J}$. Should there exist a single $j \in \mathcal{J}$ where these conditions are not satisfied, then we cannot apply the value function constraint. Therefore, we shall branch from S, generating $2n + 1$ child nodes, with the local constraints described by Figure 4.1. As

we can see, for the first $2n$ nodes, there exists a single local constraint which implies $(\tilde{y}, \tilde{y})$ is an infeasible response for the follower and therefore we do not need to apply the necessary cutting plane. Whereas at the final node, the local constraints imply that $(\tilde{y}, \tilde{y})$ is feasible and we thus can apply the cutting plane locally. However, some solvers do not allow for more than 2 branches from a single node. Therefore, to recreate Figure 4.1, we use Figure 4.2 instead, with the local constraints at each node given by

$$S_l, 1 = \begin{cases} \sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} \tilde{y}_j^{ik} - \bar{x}_j \leq 0 & \forall j \in \{1, \dots, l-1\}, \\ \sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} \tilde{y}_l^{ik} - \bar{x}_l \geq 1, \end{cases} \tag{4.11a}$$

$$S_l, 2 = \left\{ \sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} \tilde{y}_j^{ik} - \bar{x}_j \leq 0 \quad \forall j \in \{1, \dots, l\}, \right. \tag{4.11b}$$

$$S_{n+l}, 1 = \begin{cases} \sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} \tilde{y}_j^{ik} - \bar{x}_j \leq 0 & \forall j \in \{1, \dots, l\}, \\ \sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} \tilde{y}_j^{ik} - (s_j - \bar{x}_j) \leq 0 & \forall j \in \{1, \dots, l-1\}, \\ \sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} \tilde{y}_l^{ik} - (s_l - \bar{x}_l) \geq 1, \end{cases} \tag{4.11c}$$

$$S_{n+l}, 2 = \begin{cases} \sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} \tilde{y}_j^{ik} - \bar{x}_j \leq 0 & \forall j \in \{1, \dots, l\}, \\ \sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} \tilde{y}_j^{ik} - (s_j - \bar{x}_j) \leq 0 & \forall j \in \{1, \dots, l\}, \\ (4.12), \end{cases} \tag{4.11d}$$

where

$$\sum_{j\in\mathcal{J}} \Big(\sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} \bar{y}_j^{ik}(x_j + c_j) + \sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} y_j^{ik} c_j \Big) \leq \sum_{j\in\mathcal{J}} \Big(\sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} \tilde{y}_j^{ik}(x_j + c_j) + \sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} \tilde{y}_j^{ik} c_j \Big) \tag{4.12}$$

is the value function constraint. As we can see, at a node $S_l, 1$ for some $l \in \{1, \dots, 2n\}$, there exists a local constraint that implies the solution $(\tilde{y}, \tilde{y})$ is invalid and these nodes correspond the node l in Figure 4.1. The nodes $S_l, 2$ for some $l \in \{1, \dots, 2n-1\}$ can be thought of as dummy nodes, which are needed to perform this branching strategy, should we be using a solver which requires a maximum of two branches when branching. The final node $S_{2n}, 2$ is the only node where the solution $(\tilde{y}, \tilde{y})$ is locally feasible and therefore is the node where we enforce the constraint (4.12).

This branching strategy is performed at every node of the tree where we discover an integer bilevel infeasible solution, which will result in a large number of nodes being generated, which could affect the computational performance. Thus, we propose an improved strategy with the aim of generating fewer nodes.

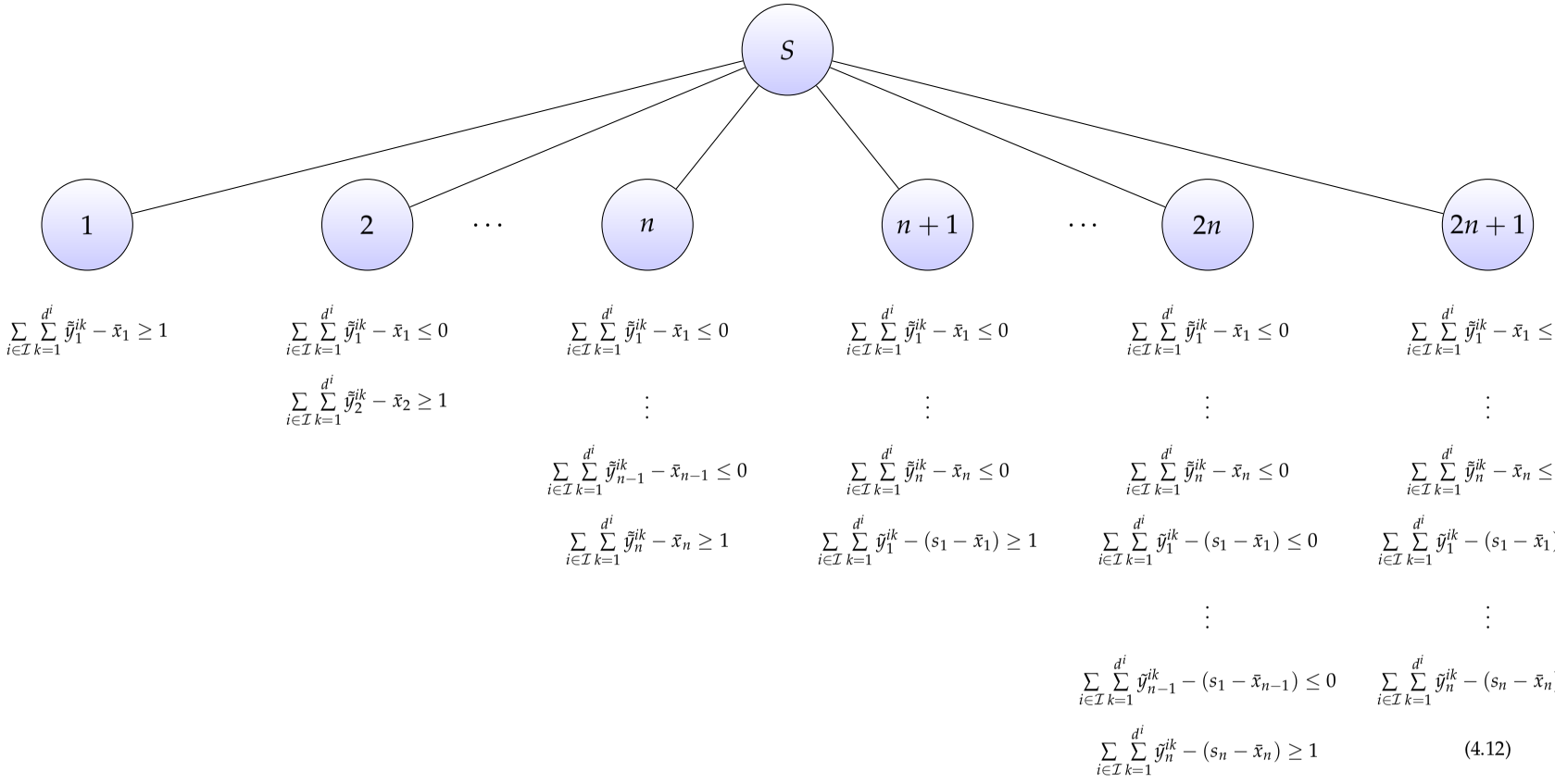


FIGURE 4.1: n-ary branching strategy

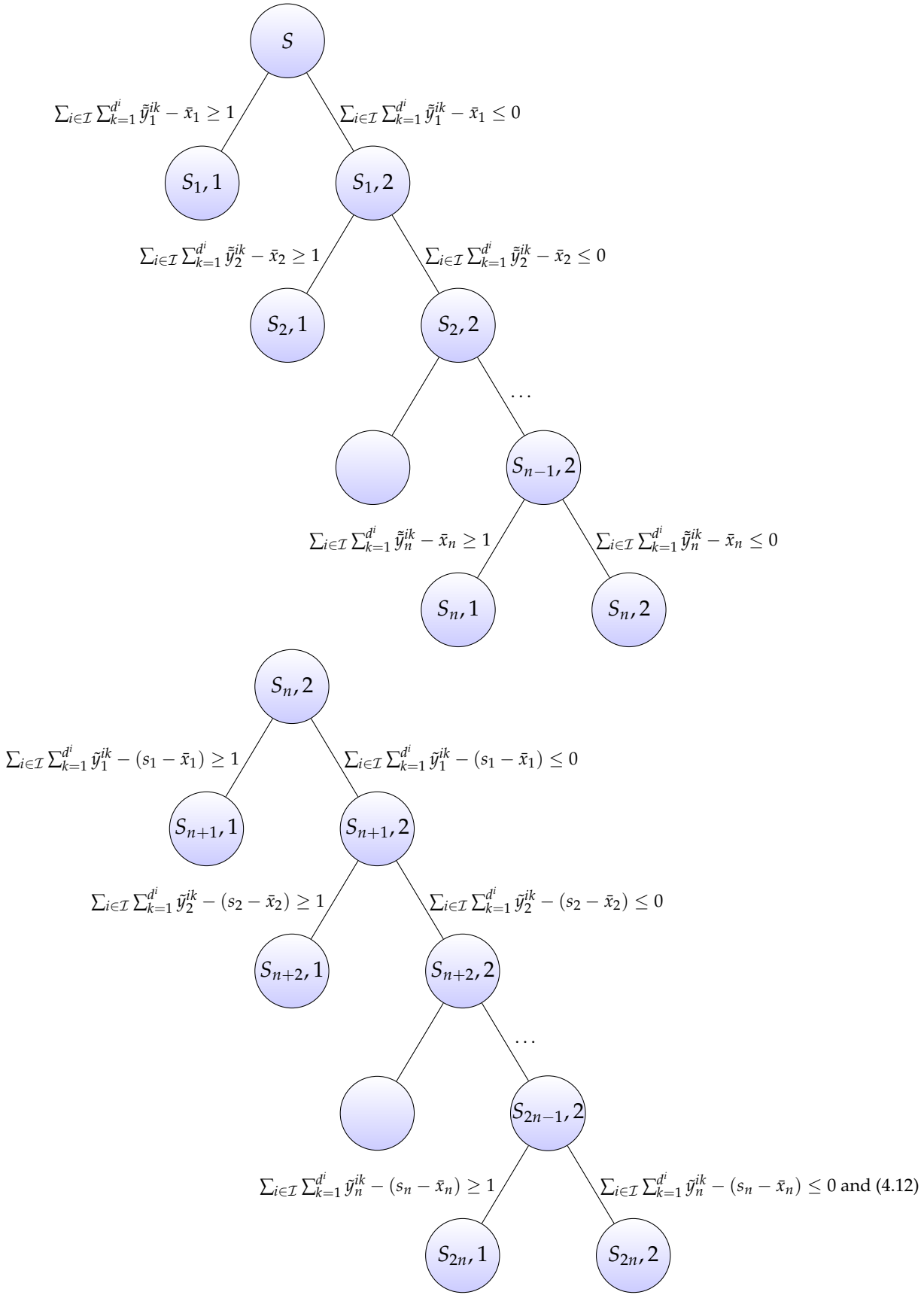


FIGURE 4.2: n-ary branching for when we can have at most 2 branches from a single node.

4.2.4 Improved n-ary Branching

Although correct, the n-ary Branching method could unnecessarily generate a large number of nodes which shall always be infeasible. For example, assume that for the solution $(\tilde{\bar{y}}, \tilde{y})$, there exists some $j \in \mathcal{J}$ such that $\sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} \tilde{\bar{y}}_j^{ik} = 0$, i.e., commodity j is not purchased from the leader. Using the n-ary Branching, we would need to generate two nodes, with the local constraints $-\bar{x}_j \geq 1$ and $-\bar{x}_j \leq 0$ respectively. As $\bar{x} \geq 0$, then the node with the constraint $-\bar{x}_j \geq 1$ shall always be infeasible and the node with the constraint $-\bar{x}_j \leq 0$ shall always be feasible. Thus, this branch was unnecessary. Likewise, for any $j \in \mathcal{J}$ such that $\sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} \tilde{y}_j^{ik} = 0$, the local constraints are $-(s_j - \bar{x}_j) \geq 1$ and $-(s_j - \bar{x}_j) \leq 0$, which shall always be infeasible and feasible, respectively. Therefore, rather than performing the n-ary Branching for every commodity, we would only branch on commodities purchased by the follower, i.e., $\bar{x}_j > 0$. In doing this, we should hopefully reduce the number of nodes generated at every bilevel infeasible solution and thus reduce the computational time and expense.

4.3 "Pre-Computed Follower Solutions" Methods

Thus far, the solution methods presented have been general and have not assumed the leader has any extra information other than the follower's objective and feasible region. Should the leader have already pre-computed a subset of follower points then intuitively we want to take advantage of this information. In the following, we present formulations and solution methods which utilize any pre-computed follower points. We shall focus our attention to the non-unit supply case, however, the relevant unit supply formulations can be found in Appendix A.

4.3.1 Single MINLBP Method

For the single MINLBP method, we shall be assuming that all of the follower's feasible points have been pre-computed and are available to the leader. We begin by defining $\mathcal{Y}^{Sol}$ as a subset of all feasible points for the follower, i.e.,

$$\begin{aligned} \mathcal{Y}^{Sol} \subseteq \{y : & A^i y^{ik} \leq b^i \quad \forall i \in \mathcal{I}, \forall k \in \{1, \ldots, d^i\}, \\ & \sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} y_j^{ik} \leq s_j \quad \forall j \in \mathcal{J}, \\ & y^{ik} \in \{0,1\}^n\}, \end{aligned} \tag{4.13}$$

where we assume that the set $\mathcal{Y}^{Sol}$ only contains points that are unique from the perspective of the leader, i.e.,

$$\sum_{i\in\mathcal{I}}\sum_{k=1}^{d^i} y_j^{ik} = \sum_{i\in\mathcal{I}}\sum_{k=1}^{d^i} {y'}_j^{ik} \quad \forall j \in \mathcal{J} \iff y = y' \quad \forall y, y' \in \mathcal{Y}^{Sol}. \tag{4.14}$$

By making this assumption we can reduce the size of $\mathcal{Y}^{Sol}$ without compromising the validity of the method. This is because the leader is only interested in which commodities are bought by the leaders and not by how they are distributed. I.e. a follower reaction where follower k^1 purchases commodity j and another reaction where k^2 purchases commodity j are the same in the eyes of the leader because all they care about is that commodity j was bought by somebody.

Should $\mathcal{Y}^{Sol}$ be known, then the follower's problem can be re-defined, such that rather than optimising y, they optimise the binary vector λ, which determines which point from $\mathcal{Y}^{Sol}$ they respond with. As such, the non-unit supply bilevel problem can be formulated as

$$\max_{x,\bar{x},\lambda} \quad \sum_{l=1}^{|\mathcal{Y}^{Sol}|} \lambda_l \Big(\sum_{j\in\mathcal{J}} \max\{0, \sum_{i\in\mathcal{I}}\sum_{k=1}^{d^i} y_l{}_j^{ik} - (s_j - \bar{x}_j)\}(x_j + c_j)\Big) - \sum_{j\in\mathcal{J}} \bar{x}_j c_j \tag{4.15a}$$

$$\text{s.t.} \quad (3.4b) - (3.4f), \tag{4.15b}$$

$$\lambda \in \arg\min_{\lambda}\{ \sum_{l=1}^{|\mathcal{Y}^{Sol}|} \lambda_l \sum_{j\in\mathcal{J}} \Big(\max\{0, \sum_{i\in\mathcal{I}}\sum_{k=1}^{d^i} y_l{}_j^{ik} - (s_j - \bar{x}_j)\}x_j + \sum_{i\in\mathcal{I}}\sum_{k=1}^{d^i} y_l{}_j^{ik} c_j \Big) : \tag{4.15c}$$

$$\sum_{l=1}^{|\mathcal{Y}^{Sol}|} \lambda_l = 1, \tag{4.15d}$$

$$\lambda \in \{0,1\}^{|\mathcal{Y}^{Sol}|}\}, \tag{4.15e}$$

where $y_l \in \mathcal{Y}^{Sol}$. The variable λ now indicates which one of the solutions in $\mathcal{Y}^{Sol}$ the follower's responds with, with $\lambda_l = 1$ if and only if the follower's reacts with y_l. Constraint (4.15d) ensures that the follower must select exactly 1 response from $\mathcal{Y}^{Sol}$.

To handle the max operator, we shall use a similar method to the Max Value formulation by introducing the binary variables γ such that

$$\gamma_{lj} = 1 \rightarrow \sum_{i \in \mathcal{I}} \sum_{k=1}^{d^i} y_l{}_j^{ik} \geq s_j - \bar{x}_j, \tag{4.16a}$$

$$\gamma_{lj} = 0 \rightarrow \sum_{i \in \mathcal{I}} \sum_{k=1}^{d^i} y_l{}_j^{ik} \leq s_j - \bar{x}_j, \tag{4.16b}$$

which can be enforced by using the constraints

$$\bar{M}(1 - \gamma_{lj}) \geq -\Big(\sum_{i \in \mathcal{I}} \sum_{k=1}^{d^i} y_l{}_j^{ik} - (s_j - \bar{x}_j) \Big), \tag{4.17a}$$

$$\bar{M}\gamma_{lj} \geq \sum_{i \in \mathcal{I}} \sum_{k=1}^{d^i} y_l{}_j^{ik} - (s_j - \bar{x}_j), \tag{4.17b}$$

for all $l \in \{1, \ldots, |\mathcal{Y}^{Sol}|\}$, $j \in \mathcal{J}$. The value of $\bar{M}$ can be calculated similar to that in (4.5) by defining $\bar{M} = \max\{s_j : j \in 1, \ldots, n\} + 1$. Once again, when $\sum_{i \in \mathcal{I}} \sum_{k=1}^{d^i} y_l{}_j^{ik} = (s_j - \bar{x}_j)$, γ can be either 0 or 1. We then introduce the variable z, for which we define $z_{lj} = \max\{0, \sum_{i \in \mathcal{I}} \sum_{k=1}^{d^i} y_l{}_j^{ik} - (s_j - \bar{x}_j)\}$ by using the constraints

$$z_{lj} \geq 0, \tag{4.18a}$$

$$z_{lj} \geq \sum_{i \in \mathcal{I}} \sum_{k=1}^{d^i} y_l{}_j^{ik} - (s_j - \bar{x}_j), \tag{4.18b}$$

$$z_{lj} \leq s_j \gamma_{lj}, \tag{4.18c}$$

$$z_{lj} \leq \sum_{i \in \mathcal{I}} \sum_{k=1}^{d^i} y_l{}_j^{ik} - (s_j - \bar{x}_j) + s_j(1 - \gamma_{lj}). \tag{4.18d}$$

The constraints (4.17) and (4.18) only contain leader variables, thus are considered leaders constraints and as such they shall define the leaders feasible region. Hence, the bilevel formulation becomes

$$\max_{x,\bar{x},\gamma,z,\lambda} \quad \sum_{l=1}^{|\mathcal{Y}^{Sol}|} \lambda_l \Big(\sum_{j\in\mathcal{J}} z_{lj}(x_j + c_j) \Big) - \sum_{j\in\mathcal{J}} \bar{x}_j c_j \tag{4.19a}$$

$$\text{s.t.} \quad (3.4b) - (3.4f), \tag{4.19b}$$

$$(4.17) - (4.18) \quad \forall l \in \{1,\dots,|\mathcal{Y}^{Sol}|\},\ \forall j \in \mathcal{J}, \tag{4.19c}$$

$$\gamma_l \in \{0,1\}^n \quad \forall l \in \{1,\dots,|\mathcal{Y}^{Sol}|\}, \tag{4.19d}$$

$$z_l \in \mathbb{Z}^n \quad \forall l \in \{1,\dots,|\mathcal{Y}^{Sol}|\}, \tag{4.19e}$$

$$\lambda \in \arg\min_{\lambda} \{ \sum_{l=1}^{|\mathcal{Y}^{Sol}|} \lambda_l \Big(\sum_{j\in\mathcal{J}} \Big(z_{lj} x_j + \sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} y_{l_j^{ik}} c_j \Big) \Big) : \tag{4.19f}$$

$$\sum_{l=1}^{|\mathcal{Y}^{Sol}|} \lambda_l = 1, \tag{4.19g}$$

$$\lambda \in \{0,1\}^{|\mathcal{Y}^{Sol}|} \}. \tag{4.19h}$$

In previous single level reformulations, we have been unable to use approaches such as KKT-conditions, or strong duality, as a result of the integrality constraints in the lower level. However, if we can show that all the vertices of the follower's feasible region satisfy the integrality constraints, then we can relax the follower's problem to just an LP and use the relevant reformulation techniques. From [10] we get the following theorem.

Theorem 4.2. *Let A be an integer matrix. The following statements are equivalent:*

1. *A is totally unimodular.*
2. *The extreme points (if any) of $S(b) = \{x : Ax \leq b, x \leq 0\}$ are integer for arbitrary b.*
3. *Every square nonsingular submatrix of A has an integer inverse.*

Thus, if we can show that the constraint matrix for the follower problem is totally unimodular, then we have shown that the extreme points are integer, meaning we can relax the integrality constraints. As such, we can reduce the bilevel problem to a single level by using optimality conditions used for continuous problems, such as KKT and strong duality, rather than just the value function approach. Rewriting the follower's problem in standard form, we get

$$\min_{\lambda} \quad d_1\lambda_1 + \cdots + d_{|\mathcal{Y}^{Sol}|}\lambda_{|\mathcal{Y}^{Sol}|} \tag{4.20a}$$

$$\text{s.t.} \quad \lambda_1 + \cdots + \lambda_{|\mathcal{Y}^{Sol}|} - 1 = 0, \tag{4.20b}$$

$$\lambda_i \geq 0 \quad \forall i \in \{1,\dots,|\mathcal{Y}^{Sol}|\}, \tag{4.20c}$$

where $d_l = \left(\sum_{j \in \mathcal{J}} \left(z_{lj} x_j + \sum_{i \in \mathcal{I}} \sum_{k=1}^{d^i} y_{l_j^{ik}} c_j \right) \right)$ and λ is the reaction of the follower, such that if the follower reacts with solution i then $\lambda_i = 0$. Note, $\lambda \in [0,1]^{|\mathcal{Y}^{Sol}|}$ but we have not explicitly stated λ's upper bound in (4.20). This is because (4.20b) automatically enforces an upper bound of 1 on all λ. Should there exist some $\lambda_i > 1$, then (4.20b) would be violated.

We can quickly and easily show that the constraint matrix for this problem is totally unimodular by using the consecuative one's property [87].

Theorem 4.3. *(Consecutive One's Property) If A is a 0-1 matrix, in which every row, the 1's appear consecutively, then A is totally unimodular.*

This is clearly the case for (4.20), thus we can use KKT, duality and value function approaches to produce the single level formulation

4.3.1.1 KKT Reformulation

To carry out a KKT reformulation, we must first define the Lagrangian function. The $g(x) \leq 0$ constraints are given by $-\lambda_i \leq 0$ and our single $h(x) = 0$ constraint is given by $\lambda_1 + \cdots + \lambda_{||} - 1 = 0$. So the Lagrangian is given by

$$\begin{aligned} L(\lambda, \mu, \nu) =& d_1\lambda_1 + \cdots + d_{|\mathcal{Y}^{Sol}|}\lambda_{|\mathcal{Y}^{Sol}|} + \mu_1(-\lambda_1) + \cdots + \\ & \mu_{|\mathcal{Y}^{Sol}|}(-\lambda_{|\mathcal{Y}^{Sol}|}) + \nu(\lambda_1 + \cdots + \lambda_{|\mathcal{Y}^{Sol}|} - 1), \end{aligned} \tag{4.21}$$

which produces the following KKT conditions

$$0 = d_l - \mu_l + \nu \quad \forall l \in \{1, \ldots, |\mathcal{Y}^{Sol}|\}, \tag{4.22a}$$
$$0 = \mu_l \lambda_l \quad \forall l \in \{1, \ldots, |\mathcal{Y}^{Sol}|\}, \tag{4.22b}$$
$$0 = \lambda_1 + \cdots + \lambda_{|\mathcal{Y}^{Sol}|} - 1, \tag{4.22c}$$
$$0 \leq \lambda, \mu, \tag{4.22d}$$
$$\nu \quad \text{free}, \tag{4.22e}$$

resulting in the KKT formulation to be

$$\max_{x,\bar{x},\lambda,\mu,\nu} \quad \sum_{l=1}^{|\mathcal{Y}^{Sol}|} \lambda_l \Big(\sum_{j\in\mathcal{J}} z_{lj}(x_j + c_j) \Big) - \sum_{j\in\mathcal{J}} \bar{x}_j c_j \tag{4.23a}$$

$$\text{s.t.} \quad \text{(3.4b)} - \text{(3.4f)}, \tag{4.23b}$$

$$\text{(4.17), (4.18)} \quad \forall l \in \{1,\dots,|\mathcal{Y}^{Sol}|\},\ \forall j \in \mathcal{J}, \tag{4.23c}$$

$$0 = \sum_{j\in\mathcal{J}} \Big(z_{lj}x_j + \sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} y_{lj}^{ik} c_j \Big) - \mu_l + \nu \quad \forall l \in \{1,\dots,|\mathcal{Y}^{Sol}|\}, \tag{4.23d}$$

$$0 = \mu_l \lambda_l \quad \forall l \in \{1,\dots,|\mathcal{Y}^{Sol}|\}, \tag{4.23e}$$

$$0 = \lambda_1 + \cdots + \lambda_{|\mathcal{Y}^{Sol}|} - 1, \tag{4.23f}$$

$$\lambda, \mu \geq 0, \tag{4.23g}$$

$$\nu \text{ free}. \tag{4.23h}$$

This formulation is non-linear, however, we can linearise most of it by using the McCormick and binary expansion techniques that have been described previously. In (4.23) we have the quadratic terms $z_{lj}x_j$, $\lambda_l z_{lj}$ and $\mu_l\lambda_l$ and the cubic term $\lambda_l z_{lj}x_j$. As z is an integer variable and x_j is a continuous one, we can represent their product using the McCormick constraints, after performing a binary expansion on z to get the binary variable α and defining $p_{ljm} = \alpha_{ljm}x_j$. For $\lambda_l z_{lj}$, we shall define $r_{lj} = \lambda_l z_{lj}$ and use the McCormick constraints directly as λ is binary. Following this, $\lambda_l z_{lj}x_j$ can be written as the product between λ_l and $\sum_{m=1}^{\lfloor \log_2 z_{lj} \rfloor} 2^{m-1} p_{ljm}$, which, once again, can be linearised using the McCormick constraints, as λ is a binary variable and p is continuous, defining $q_{ljm} = \lambda_l p_{ljm}$. Linearising $\mu_l\lambda_l$ is much trickier as μ_l is continuous and does not have an upper bound. Work can be done to create upper and lower bounds for this variable, by analysing the dual problem, or we could use a relaxation method that utilises NCP-functions [150]. However, with some state-of-the-art solvers, such as CPLEX, we can define variables as a Special Ordered Set (SOS) Type 1, which implies that at most one variable can be non-zero. The solver then implements a special branching strategy which limits the number of non-zero elements. By defining $|\mathcal{Y}^{Sol}|$-many SOS Type 1 constraints, we will have enforced the complementarity constraints of the KKT conditions. The linear KKT formulations for the unit and non-unit supplies are given by (A.3) and (A.8) respectively.

4.3.1.2 Value Function Reformulation

As discussed in Chapter 2, the value function reformulations moves the follower's variables and constraints into the leader's problem, whilst simultaneously introducing additional constraints that force the follower's variables to represent the followers true

reaction. Given that $\mathcal{Y}^{Sol}$ is known, we can enforce the value function constraint by using

$$\sum_{l=1}^{|\mathcal{Y}^{Sol}|} \lambda_l \sum_{j\in\mathcal{J}} (z_{lj}x_j + \sum_{i\in\mathcal{I}}\sum_{k=1}^{d^i} y_{l_j}^{ik} c_j) \leq \sum_{j\in\mathcal{J}} (z_{oj}x_j + \sum_{i\in\mathcal{I}}\sum_{k=1}^{d^i} y_{o_j}^{ik} c_j) \quad \forall o \in \{1,\dots,|\mathcal{Y}^{Sol}|\}. \tag{4.24}$$

Similar to before, we still have the quadratic terms $z_{lj}x_j$, $\lambda_l z_{lj}$ and the cubic terms $\lambda_l z_{lj} x_j$, which are linearised using the same combinations of binary expansions and McCormick constraints. The linear value function formulations for the unit and non-unit supplies are given by (A.4) and (A.9) respectively.

4.3.1.3 Strong Duality Reformulation

As we have shown, we can use the KKT conditions to transform the bilevel problem to a single level because we reformulated the follower's problem, such that it is totally unimodular. Likewise, because of this unimodularity, we can use the strong duality conditions. Given the linear program (4.20), the corresponding dual is given by

$$\max_{\pi} \quad \pi \tag{4.25a}$$
$$\text{s.t.} \quad \pi \leq d_i \quad \forall i \in \{1,\dots,|\mathcal{Y}^{Sol}|\}, \tag{4.25b}$$
$$\pi \text{ free.} \tag{4.25c}$$

The strong-duality conditions that would be used to ensure follower optimality are given by

$$\sum_{l=1}^{|\mathcal{Y}^{Sol}|} \lambda_l \sum_{j\in\mathcal{J}} (z_{lj}x_j + \sum_{i\in\mathcal{I}}\sum_{k=1}^{d^i} y_{l_j}^{ik} c_j) = \pi \tag{4.26a}$$
$$\pi \leq \sum_{j\in\mathcal{J}} (z_{oj}x_j + \sum_{i\in\mathcal{I}}\sum_{k=1}^{d^i} y_{o_j}^{ik} c_j) \quad \forall o \in \{1,\dots,|\mathcal{Y}^{Sol}|\} \tag{4.26b}$$
$$\pi \text{ free,} \tag{4.26c}$$

where (4.26a) ensures the strong-duality property and (4.26b) enforces dual feasibility. In this scenario, we can see that the constraints (4.26a) and (4.26b) can be merged, to project out π, leaving us with a constraint exactly the same as if we used the value

function approach. Thus, the duality formulation is the same as the value function formulation.

4.3.2 Double MINLBP Method

We have just demonstrated how the bilevel problem can be reduced to a single-level formulation by using the KKT, value function and strong duality conditions, when all feasible follower solutions have been computed. However, this information is seldom known and, to calculate, would be far to expensive computationally for any non-trivial problem. Therefore, we present solution methods, which use a relaxation of the presented formulations, by using a subset Y of $\mathcal{Y}^{Sol}$.

Let DM(Y) be the formulation of (4.15) where instead of using $\mathcal{Y}^{Sol}$ we use the subset Y and let DMKKT(Y) and DMVF(Y) be the single-level reformulations using the KKT and value function method respectively. Clearly, if $|Y| < |\mathcal{Y}^{Sol}|$, then DMKKT(Y) and DMVF(Y) are relaxations of the original bilevel problem and may not provide a bilevel feasible solution, let alone the optimal solution. Although these reformulations force the follower's choice of solution to be the cheapest from the set Y, there may exist a solution not within Y, which, given the leader's choice of variables, would produce a smaller objective for the follower or a larger objective for the leader. Therefore, we present the following Known Solution's Algorithm (KSA) solution method.

Algorithm KSA

1. Create an initial set $Y \subset \mathcal{Y}^{Sol}$.
2. Repeat Steps 2-6 until we discover a bilevel feasible solution.
3. Generate the formulation DMKKT(Y) or DMVF(Y) and solve to get the solution $(x^*, \bar{x}^*, \lambda^*)$ with the leader and follower objectives F^{U*} and F^{L*} respectively.
4. Solve the followers problem parameterised by $(x^*, \bar{x}^*)$ to get $\phi(x^*, \bar{x}^*)$ and $\psi(x^*, \bar{x}^*)$.
5. If $F^{L*} \leq \phi(x^*, \bar{x}^*)$ we have a bilevel feasible solution and go to step 7.
6. Else, add the solutions found in $\psi(x^*, \bar{x}^*)$ to Y and return to step 3.
7. Solve the original bilevel problem (3.4) with with the added constraints that $y \notin Y$ and $F^U \geq F^{U*}$ and, if feasible, get the solution $(x', \bar{x}', y')$.
8. If the solution to step 7 is infeasible, then the optimal leader variables are $(x^*, \bar{x}^*)$ else $(x', \bar{x}')$ is the optimal solution.

In the above algorithm, we repeatedly solve the relaxation using the set Y. If the solution to DMKKT(Y) or DMVF(Y) is not bilevel feasible, because there exists a better response for the follower, we introduce this new solution to Y and repeat the process until we discover a bilevel feasible solution. Once we have a bilevel feasible solution, we go back to the general formulation and restrict the follower to responding with a solution not from Y.

Forcing the follower to a point not in Y can be achieved by using a no-good cut. For the unit-supply case, we use the single constraint

$$\sum_{i\in\mathcal{I}}\sum_{k=1}^{d^i}\Big(\sum_{j\in\mathcal{Y}^0} y_j^{ik} + \sum_{j\in\mathcal{Y}^1}(1-y_j^{ik})\Big) \geq 1 \tag{4.27}$$

for every $y^{'} \in Y$ where $\mathcal{Y}^0 := \{j \in \mathcal{J} : \sum_{i\in\mathcal{I}}\sum_{k=1}^{d^i} {y^{'}}_j^{ik} = 0\}$ and $\mathcal{Y}^1 := \{j \in \mathcal{J} : \sum_{i\in\mathcal{I}}\sum_{k=1}^{d^i} {y^{'}}_j^{ik} = 1\}$. For the non-unit supply case, for each every $y^{'} \in Y$ we introduce n-many binary variables which are equal to 0 if $\sum_{i\in\mathcal{I}}\sum_{k=1}^{d^i} y_j^{ik} = \sum_{i\in\mathcal{I}}\sum_{k=1}^{d^i} {y^{'}}_j^{ik}$ for the given j and 1 otherwise. We then force the summation of these binary variables to be greater than 0, forcing the followers to purchase a differing amount for some commodity j.

Along with this, we place a lower bound on the leader's objective value, hoping to find a better solution for the leader. Ideally, the solution from the last iteration of DMKKT(Y) or DMVF(Y) should provide a bilevel solution, which causes the second problem to be either infeasible, or much easier to solve. As one can imagine, the number of iterations between steps 2 and 6 shall depend on both the size and contents of the initial set Y.

4.3.2.1 Y Selection

Determining Y is clearly an important process that can affect the overall performance of KSA. Selecting a large set Y will result in solving the follower's problem many times, which is computationally expensive, but can provide better solutions and stronger bounds for step 7 of the algorithm. Whereas, a small set Y will be much more manageable computationally, but may not produce enough useful information. However, its not just the size of Y that should be carefully calculated but also the contents, i.e. the feasible points for the follower. Ideally, we want to construct Y in such a manner that each of its elements have a high probability of being the optimal solution whilst keeping the size of Y relatively small. To begin with, we shall assume that all of the solutions within $\mathcal{Y}^{Sol}$ are ordered by their cost when there is no taxation, i.e.,

$$\sum_{j\in\mathcal{J}}\sum_{i\in\mathcal{I}}\sum_{k=1}^{d^i} {y_1}_j^{ik} \leq \sum_{j\in\mathcal{J}}\sum_{i\in\mathcal{I}}\sum_{k=1}^{d^i} {y_2}_j^{ik} \leq \cdots \leq \sum_{j\in\mathcal{J}}\sum_{i\in\mathcal{I}}\sum_{k=1}^{d^i} {y_{|\mathcal{Y}^{Sol}|}}_j^{ik}. \tag{4.28}$$

This leads us into the following lemmata.

Lemma 4.4. *If the leader does not purchase a commodity that belongs to the solution y_1, i.e. for all $j \in \mathcal{J}$, $\bar{x}_j = 0$ if $\sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} {y_1}_j^{ik} \geq 1$, then the maximum objective value they can achieve is 0.*

Proof. Firstly, as y_1 is the cheapest follower solution when 0 taxation is applied and all taxation is non-negative, then if no commodity belonging to y_1 has any taxation applied to it, then y_1 shall remain the cheapest solution for the follower. Thus, the follower shall always respond with y_1. Therefore, if the leader has not purchased one of these commodities, then they cannot achieve any income and thus the maximum objective value they can achieve is 0, if they purchase no commodities at all. □

Lemma 4.5. *For all $l \in \{1, \ldots, |\mathcal{Y}^{Sol}|\}$ such that*

$$\sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} {y_1}_j^{ik}(M + c_j) \leq \sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} {y_l}_j^{ik} c_j, \tag{4.29}$$

y_l will never be an optimal response for the follower.

Proof. By definition, the follower responds to the leader's action by selecting the cheapest response. $\sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} {y_1}_j^{ik}(M + c_j)$ is the maximum price that the solution y_1 can possibly have. Therefore, any solution y_l, whose minimum price, $\sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} {y_l}_j^{ik} c_j$, is strictly greater than the maximum price of y_1 will never be an optimal response for the follower.

For the solutions y_l, where $\sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} {y_1}_j^{ik}(M + c_j) = \sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} {y_l}_j^{ik} c_j$, the case may arise that the final costs for y_1 and y_l are equal. In which case, the follower shall have no preference on which solution they choose. However, we have been focusing on the optimistic bilevel setting, which means that in such a case, the leader shall assume that the follower selects the solutions that provides them with the largest objective. This corresponds to the solution y_1 as they have no income from the solution y_l, meaning the maximum objective they can achieve is 0. □

Lemma 4.6. *Following on from Lemma 4.5, for all $l \in \{1, \ldots, |\mathcal{Y}^{Sol}|\}$ such that*

$$z \leq \sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} {y_l}_j^{ik} c_j, \tag{4.30}$$

where

$$z = \max_{x,\bar{x}} \quad \sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} {y_1}_j^{ik}(x_j + c_j) \tag{4.31a}$$

$$s.t. \quad (3.4b) - (3.4f), \tag{4.31b}$$

y_l will never be an optimal response for the follower.

Proof. Similar to to Lemma 4.5, we show that any solution y_l whose cost without any taxation is greater than the maximum price of y_1, shall never be part of the optimal solution. Whereas before, we just took the price of y_1 where every commodity had maximum taxation, we now take into account the leaders budget constraints and use the maximum price that the leader could possibly make y_1 have. □

Using these Lemmas we achieve the following.

Theorem 4.7. *If $Y \subseteq \mathcal{Y}^{Sol}$ is such that for all $l \in \{1, \dots, |\mathcal{Y}^{Sol}|\}$ if $\sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} {y_l}_j^{ik} c_j \leq z$, where z is the optimal solution to* (4.31)*, then $y_l \in Y$, the optimal solution to DM(Y), is the optimal solution to original bilevel problem.*

Proof. For the set Y as described, any solution that is not part of this set will never be part of the optimal solution by Lemma 4.31. Therefore, we can solve the original bilevel problem by fixing the follower's variables to only those that appear in Y. □

In Chapter 5, Y is generated by repeatedly solving the followers problem, when no taxation is being applied, and introducing a no-good-cut. By doing this m-many times we shall generate the m cheapest follower reactions when no taxation has been applied. This means that Y contains follower points which are most likely to be affected by the leaders taxations but may not represent Y found by Theorem 4.7.

In relation to the motivation of this problem, the initial set Y may not have to be generated. Many websites, such as [1], have dedicated forums for gamers to display and compare their solutions to these SBC's. Here, users will upload feasible teams to the SBC's, which are ordered by their price, using the current value of the commodities. Although gamers can submit any team that satisfies the constraints, they tend to upload solutions with the goal of trying to find the cheapest team. Thus, taking the top m-many teams from the top can be considered as taking the m-many cheapest teams without leader taxation.

4.3.3 Known Solution Branching (KSB) Method

With the DM(Y) method, we have to solve two MINLBP's, one where we assume the follower reacts within a known set of responses and the other when their decision is not within the set. Similar to the intuition behind the n-ary Branching, once again we want to find a way in which we can contain these MILPs within a single Branch-and-Cut framework.

Let us assume that a bilevel problem is being solved with one of the solution methods presented in Section 4.1 and 4.2 and we have a known set of follower responses Y. If, at the root node, the methods from 4.1 and 4.2 decide to branch, then we shall force the solver to perform the branching outlined by Figure 4.3. We have generated $|Y| + 1$ many child nodes from the root node, with $|Y|$ many fixing the followers response and the final node allowing the follower a free choice. The first $|Y|$ nodes replicate the DM(Y) problem in step 3 of the KSA algorithm and node $S_{|Y|+1}$ acts as the MINLBP solved in step 7.

With the unit supply case, the KSB solution method may be used alongside the CP method. For the non-unit supply case, the KSB method can be used with both dichotomic and max value formulations. However, we shall only use it with the n-ary Branching method. The purpose of the KSB method is to encompass the simpler bilevel problems, where we have fixed the follower's response, within a single Branch-and-Cut framework. With the NSFCP and SFCP solution methods, additional variables need to be introduced with every infeasible bilevel solution. Therefore combining the KSB method with either the NSFCP or SFCP methods can be seen as counter intuitive.

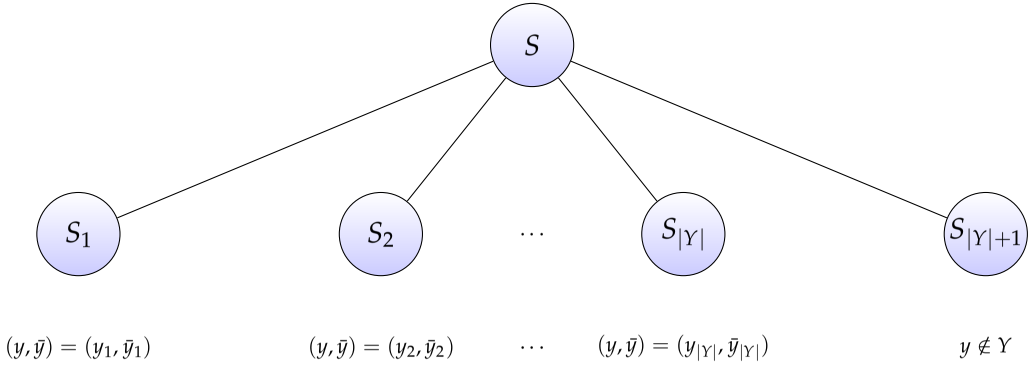


FIGURE 4.3: Branching strategy for KSB method using exact solutions.

Chapter 5

Computational Results

The primary goal of this chapter is to compare the performances of the solution methods presented in Chapter 4 along with the different formulations from Chapter 3. Our experiments are performed on a single node of the IRIDIS 5 cluster, equipped with 40 dual 2.0GHz Intel Skylake processors and 192GB of DDR4 memory, running each computation in a single thread [118]. All formulations and instances were coded in Python3.6, using CPLEX 20.10 as the solver for every HPR. As discussed in Chapter 4, we use the *LazyConstraintCallback*, *BranchCallback* and *IncumbentCallback*'s to integrate our solution methods with CPLEX.

To compare the formulations and solution methods, we used three sets of instances. In Set A, we varied the number of commodities along with the supply for each commodity. For Set B, we changed the supply for each commodity along with the number of followers and in Set C the instances were generated with varying M and B values. The results from all instances can be found at [149], along with additional plots not contained within this these.

In the remainder of this chapter, when referring to a specific formulation and solution method, we shall use (X, Y, Z) where X is the formulation, Y is the McCormick linearisation being used and Z is the solution method. Table 5.1 gives the abbreviations to be used. For example, (DI, DMC, SFCP) shall represent the dichotomic formulation with direct mcCormick linearisation used with the symmetry free cutting plane solution method.

Note, in this chapter we do not compare our solution methods directly with any other methods. This is due to a lack of algorithms being available to download and use. We did however discover one solver which was downloadable [147], which follows the work in [68, 70, 71]. This solver follows a similar method to our solution methods, using CPLEX as the main integer solver and introducing the necessary procedures to achieve a bilevel solution. When trying to use this solver for the instances we have generated,

TABLE 5.1: List of abbreviations used to distinguish between formulations, McCormick linearisation techniques and solution methods.

	Abbr	**Meaning**
Formulation	US	Unit Supply formulation
	DI	Dichotomic formulation
	MV	Max Value formulation
	DMKKT	Double MINLBP KKT Formulation
	DMVF	Double MINLBP VF Formulation
McCormick	SMC	Summation McCormick
	DMC	Direct McCormick
Solution Method	CP	Cutting plane
	KSBCP	Known Solution Branching Cutting Plane
	SFCP	Symmetry Free Cutting Plane
	NSFCP	Non-Symmetry Free Cutting Plane
	nB	n-ary Branching
	nB+	n-ary Branching plus
	KSBnB	Known Solution Branching with n-ary Branching
	KSBnB+	Known Solution Branching with n-ary Branching plus
	KSA	Known Solution Algorithm

we encountered two problems. Firstly, [147] required an older version of CPLEX, 12.7, which although is available to download, may skew the results. And secondly, but most importantly, we found that when [147] was applied to our instances it took much longer to solve than our solution methods. This was most notable where our solvers were able to solve instances in seconds and [147] was over 10 minutes without reaching the solution. As such, we have not compared our solution methods to [147], nor any other solution methods.

5.1 Results for Instance Set A

For Set A, we varied the number of commodities, n, along with the supply for each commodity, s. We allowed n to take values from the set $\{100, 125, 150, \ldots, 500\}$ and defined s as an random integer vector, where $s_j \in [1, s_{\max}]$ for $j \in \{1, \ldots, n\}$ and $s_{\max} \in \{1, 2, 4, 8\}$. M and B are both fixed to 25 along with $i = 1$ and $d^1 = 3$.

Figures 5.1, 5.3 and 5.4 demonstrate the performance of the formulations and solution methods w.r.t the computational time, the number of nodes solved and the total number of follower problems solved. In Figure 5.1 we can see that for the both Dichotomic formulations and the Max Value formulation the nB+ solution method performes best, closely followed by the KSBnB+. Across these three formulations, there was only the instance of $n = 300$ with the Dichotomic formulation and Summation McCormick's where the nB+ solution method was not the quickest.

Given these results, for the KSA solution method we used the Max Value formulation along with the nB+ solution method for the second stage. The plot, in the bottom right of Figure 5.1, shows how the KSA algorithm performed with both the KKT and VF formulations. As we can see, the KSA algorithm performed better than the other solution methods, achieving average computational times of 545.057 and 557.330 for the VF and KKT formulations respectively, compared to the (MV, DMC, nB+)'s average time of 621.898. One of the reasons behind this could be that across the 340 instances performed with the KSA algorithm, the second stage was infeasible for 304 and 305 of these instances for the VF and KKT formulations respectively. Meaning the optimal solution was discovered in the DM(Y) problem in 90% of the instances.

Figure 5.2 presents the performance profile for each solution method and formulation. With these plots, we have presented the ratio of problems solved within a given time, for each formulation and solution method. Here, we can see that the nB+ solution method is consistently solving a greater proportion of problems than the other solution methods across the Dichotomic and Max Value formulations, solving 92.6% and 92.4% within the 3600s time limit. With the Double MINLBP formulations, we can see that the KKT and VF formulations solve a similar ratio of problems within a given time. The VF formulation manages to solve 92.6% of the instances within the time limit, with the KKT formulation performing slightly better and solving 92.9%.

As expected, in Figure 5.3 we can see that although the nB+ and KSBnB+ solution methods are the best w.r.t the computational time, they produce more nodes than both the NSFCP and SFCP solution methods, which is expected, given that the solution methods force the solver to branch. Likewise, we can see that the nB and KSBnB produce significantly more nodes than all other formulations.

Figure 5.4 provides the number of times the follower's problem is solved with each solution method and formulation. Surprisingly, we can see that the NSFCP and SFCP

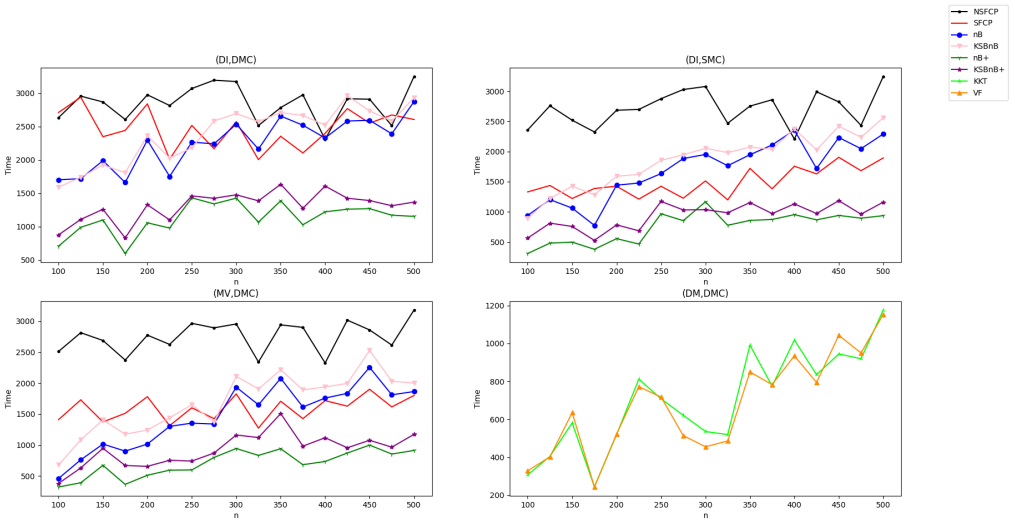


FIGURE 5.1: Set A results w.r.t computational time.

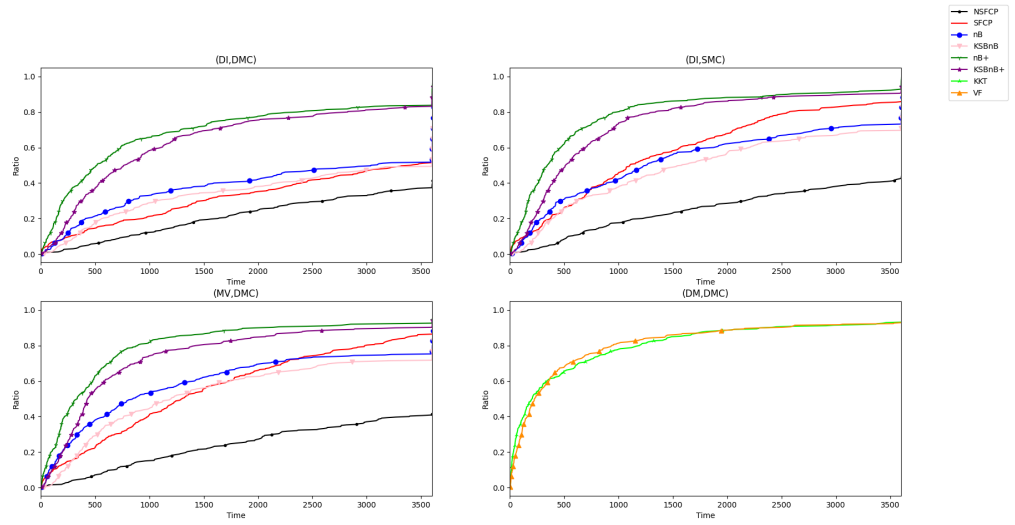


FIGURE 5.2: Proportion of instances solved within a given time for Set A.

solution methods solve the follower's problem fewer times than the branching methods for each formulation. As a result, we may find that for problems where the follower's problem is far more computationally expensive, the NSFCP and SFCP solution methods may provide optimal solutions quicker.

Note, for the solution methods that use KSB, we allow the solver to branch from the root node with $1, 2, 4$ and 8 solutions. Table 5.2 provides the average times for each KSB solution method, with each formulation. As we can see, the timings are very close, however across all instances, when the number of KSB solutions equals 1 we achieve the quickest times. Thus, in Figures 5.1–5.4 only the results where the number of KSB solutions equals 1 are shown.

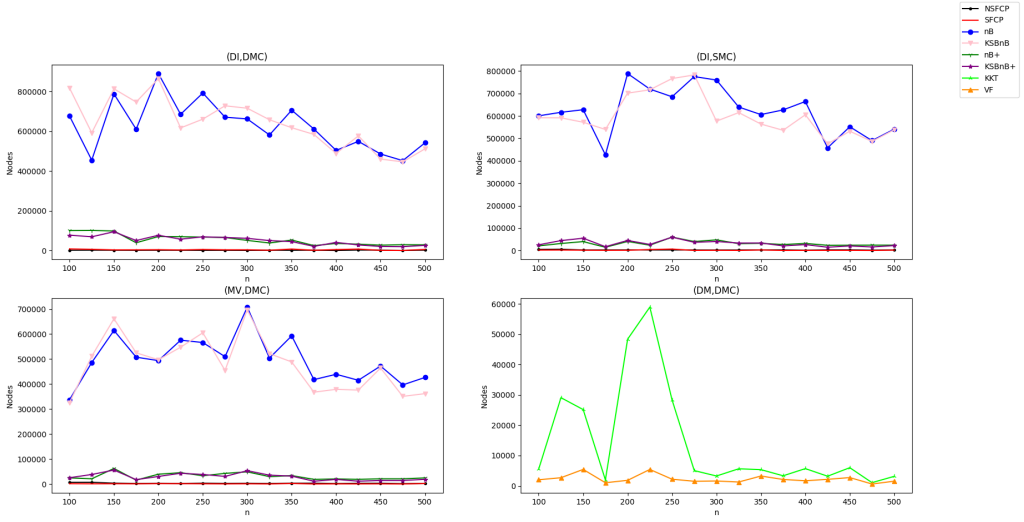


FIGURE 5.3: Set A results w.r.t the number of nodes solved.

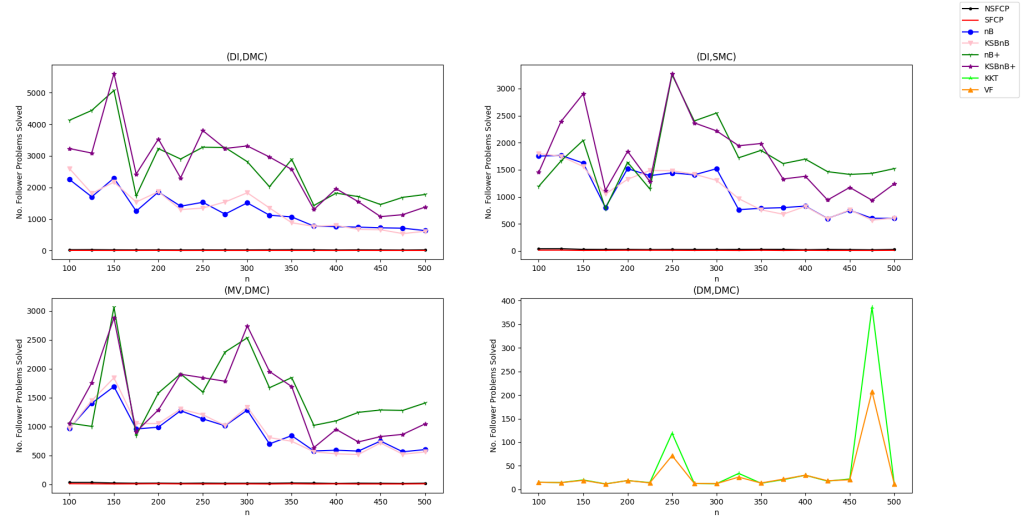


FIGURE 5.4: Set A results w.r.t the number of times the followers problem is solved.

With Set A, we also compared the performances of the unit supply formulation and solution methods. Alongside comparing them with each other, we can also measure their performances with the non-unit supply formulations and solutions methods when the supply for each commodity was set to 1.

Figure 5.5 presents the computational times for the unit supply instances. Similar to the non-unit supply cases, we find that the nB+ and KSBnB+ solution methods perform best with the Dichotomic and Max Value formulations and both KKT and VF formulations perform well compared to the other solution methods. However, these results are dwarfed by the unit supply formulation and solution methods.

Note, in Figure 5.5 the y-axis for the non-unit supply methods ranges between 0 and

TABLE 5.2: Average time for the KSB solution methods within Set A.

			No. KSB Solutions			
Formulation	**McCormick**	**Algorithm**	**1**	**2**	**4**	**8**
DI	**DMC**	**KSBnB**	2387	2433	2422	2445
		KSBnB+	1310	1350	1389	1400
	SMC	**KSBnB**	1858	1883	1906	1918
		KSBnB+	935	949	948	973
MV	**DMC**	**KSBnB**	1686	1724	1744	1754
		KSBnB+	923	951	943	972

3600, where as the (US, DMC), (US, SMC) and (DM, DMC) formulations range between 0 and 150, 25 and 25 respectively. As we can see, both the CP and KSBCP solution methods with the (US, SMC) formulation massively outperform all other solution methods, taking on average less than 5 seconds across all instances. Both solution methods behave similarly, with the CP method performing better in just the $n = 300, 325, 350$ and 425 instances.

With the Double MINLBP formulation, we used the (US, SMC) formulation with the CP solution method for the second stage of the KSA method. The timings here are not as good as the (US, SMC, CP) times, however, we find that the VF formulation performed much better than the KKT formulation.

FIGURE 5.5: Set A results w.r.t the computational time with the unit supply instances.

5.2 Results for Instance Set B

For the instances in Set B, we shall vary the supply of each commodity along with the number of followers. i shall remain fixed to 1, however we shall allow for $d \in \{5, 10, 15, 20, 25\}$. s shall be an integer vector, where $s = \lfloor [(2^{\lfloor \log_2(d+1) \rfloor} - 1)s_{\text{coef}}] \rfloor$ where $s_{\text{coef}} \in \{0.5, 0.55, \ldots, 1.5\}$. The follower's combinatorial constraints are the same as described in Set A and the parameters n, M and B shall be fixed to 200, 25 and 25 respectively. In doing so, we should be able to compare the performances of the Direct and Summation McCormick methods and see if they coincide with our predictions from Section 3.7.2.1.

Figures 5.6 and 5.7 show how the solution methods perform, with respect to computational time, for varying values of d and s_{coef}, respectively, with Figure 5.8 providing a performance plot across all instances. In this set of instances, we can see that the timings for each solution method are a lot more compact than with Set A. For the Dichotomic Direct McCormick and Max Value formulations we still have that the nB+ solution methods provides the least computational time. However with the Dichotomic Summation McCormick we find that in some instances the SFCP solution method performs best.

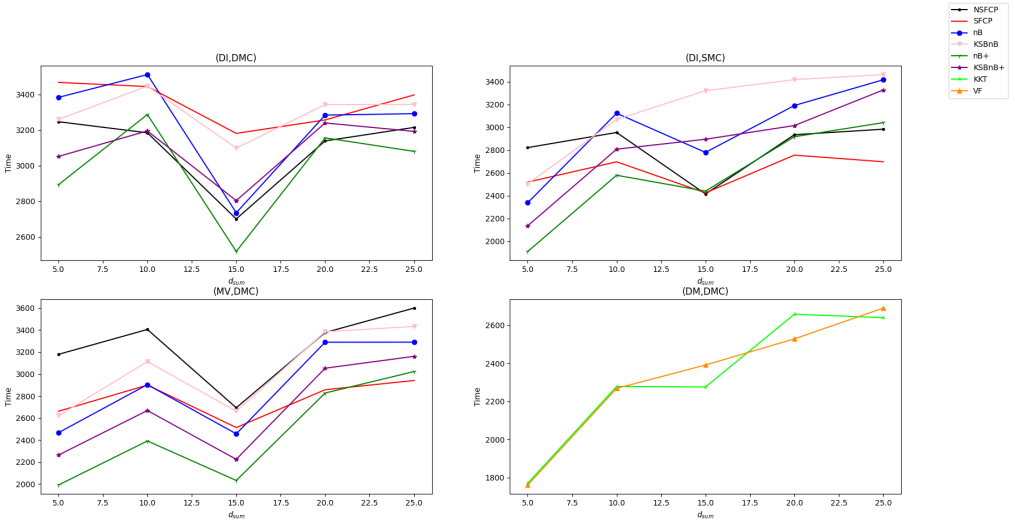


FIGURE 5.6: Set B results for d_{sum} vs Time.

Once again though, we find that the KSA solution method with the Double MINLBP formulations outperform the solution methods used with the Dichotomic and Max Value formulations. The Value Function formulation achieved an average time of 2327.641, with the KKT formulation improving on this with a time of 2324.355. Both of which were faster than using nB+ with the Max Value formulation, the quickest method from the Dichotomic and Max Value formulations.

Figure 5.8 presents the proportion of instances solved within a given time. The (DI, SMC, SFCP), (DM, DMC, KKT) and (DM, DMC, VF) perform best, solving 45.1%, 45.3% and 45.5% of all instances within the time limit, respectively. From the Dichotomic Direct McCormick formulation the NSFCP solution method solved the greatest proportion of instances, however it was only able to solve 24% of instances, significantly less than the other formulations. Most of these performance plots appear to flatten as we approach the time limit, apart from (DI,SMC,SFCP). This indicates that if we increase the time limit then we may find that (DI,SMC,SFCP) solves a larger proportion of instances than the other formulations and solution methods.

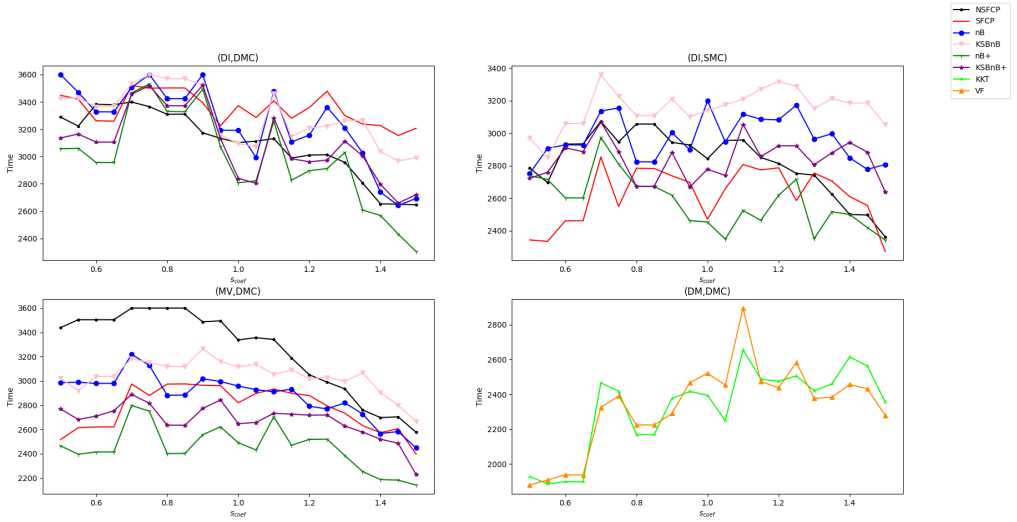


FIGURE 5.7: Set B results for s_{coef} vs Time.

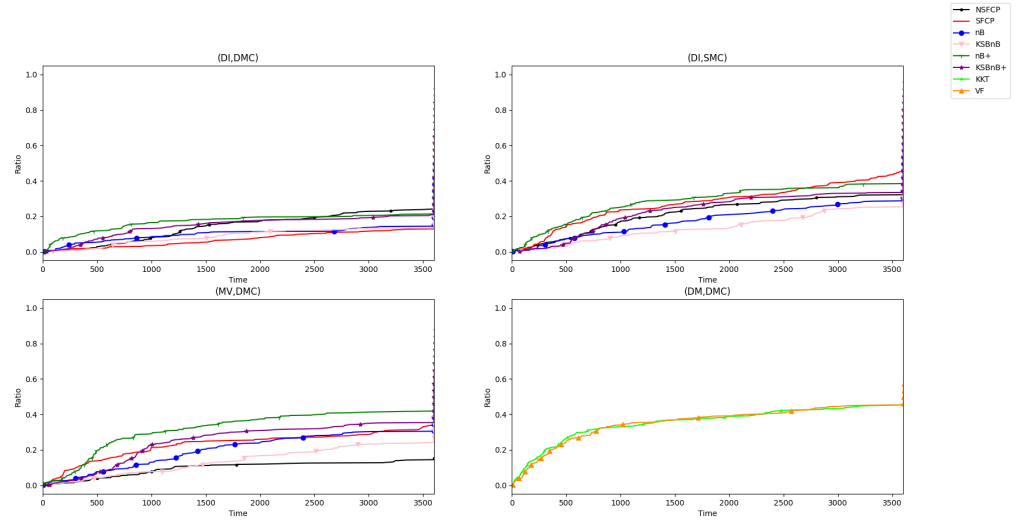


FIGURE 5.8: Proportion of instances solved within a given time for Set B.

Figure 5.9 presents the computational times for the solution methods when used with the Direct and Summation McCormicks. From Section 3.7.2.1, we predicted that when

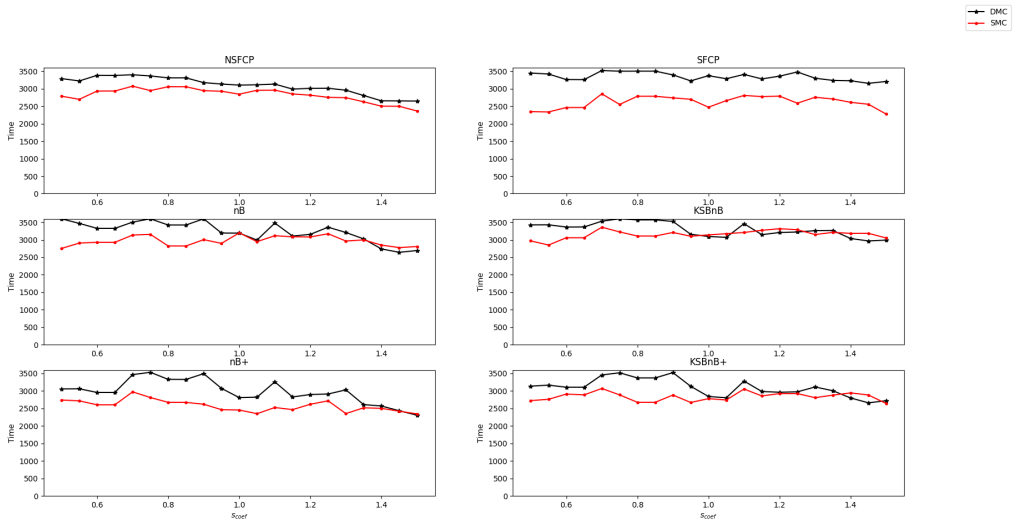


FIGURE 5.9: Set B results for SMC and DMC formulations.

$s < 2^{\lfloor \log_2(m+1) \rfloor} - 1$, the SMC formulations should reach the solution quicker, when $s > 2^{\lfloor \log_2(m+1) \rfloor} - 1$, the DMC formulations should reach a solution quicker and at equality they should perform the same. For this set of instances, these three situations correspond to when $s_{\text{coef}} = \{0.5, 0.75\}$, $s_{\text{coef}} = \{1.25, 1.5\}$ and $s_{\text{coef}} = 1$ respectively. Thus, we would expect the lines in Figure 5.9 to cross over at the point $s_{\text{coef}} = 1$. Note, for this comparison only the Dichotomic formulations have been used. Although the Max Value and Double MINLBP formulations use the Direct McCormick formulations, we cannot use the Summation McCormick and have thus omitted these from this comparison.

For the SFCP and NSFCP solution methods we can see that the Summation McCormick formulation is consistently faster than the Direct, achieving a smaller computational time for all values of s_{coef}. A similar outcome occurs with the nB+ solution method, with the Summation McCormick performing quicker for all values of s_{coef} except $s_{\text{coef}} = 1.5$, where the Direct McCormick method is marginally faster. For the remaining solution methods, we can spot a general pattern. For $s_{\text{coef}} \leq 0.95$, the Summation McCormick is faster and for $s_{\text{coef}} \geq 1$ the times become very similar, with the nB, KSBnB, and KSBnB+ solution methods having average times, for $s_{\text{coef}} \geq 1$, of 3054.300, 3157.000 and 2921.793 and 2999.925, 3198.784 and 2856.874 for the Direct and Summation McCormicks respectively.

These results contradict our predictions from Section 3.7.2.1. As expected the Summation McCornicks are faster for values of $s_{\text{coef}} < 1$. However they tend to still be faster for $s_{\text{coef}} \geq 1$, with the difference between the Direct and Summations becoming smaller.

5.3 Results for Instance Set C

For the instances in Set C, we varied the maximum taxation the leader could apply, M along with the budget of the leader, B. As such, we allowed for both M and B to take values from $\{10, 15, 20, \ldots, 75\}$. For the other parameters we have $n = 200$, $i = 1$, $d^1 = 3$ and the supply for each commodity took the same approach as in Set A, with $s_{\max}$ fixed to 8.

Figures 5.10 and 5.11 show how the solution methods perform with each formulation as the values of B and M changed respectively, with respect to time. In Figure 5.10 we can see that, generally, as the leader's budget increases, so does the computational time. There appears to be a significant increase in the computational time between $B = 35$ and 40. The most likely reasoning for this "step" is that the possible commodities that the leader could purchase with a budget of 40 is far greater than with a budget of 35. For values of $B > 40$, we can see that the times flatten out, implying that the commodities that the leader could purchase with a budget greater than 40 is the roughly the same than at 40.

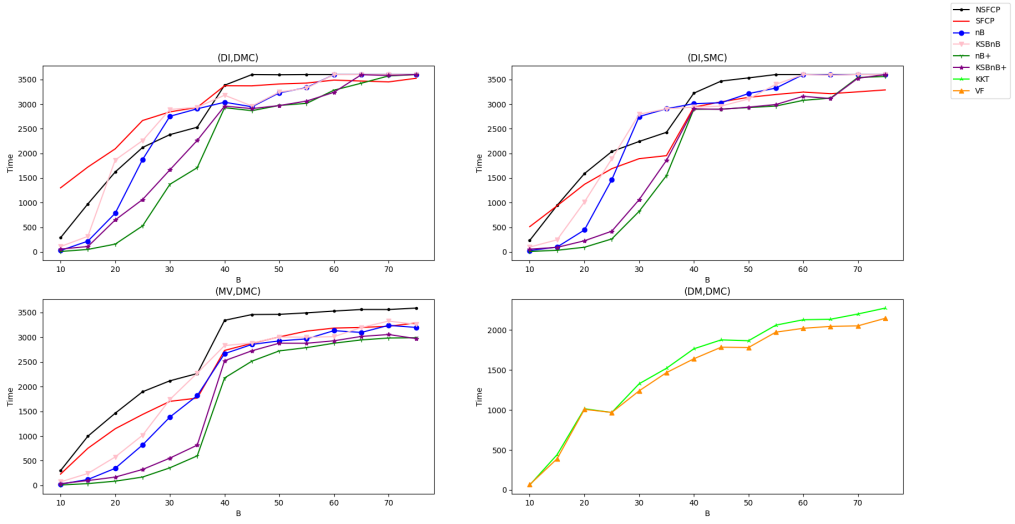


FIGURE 5.10: Set C results as the leaders budget varied.

In the dichotomic and max value formulations we find that generally the nB+ solution methods performs best, closely followed by the KSBnB+ method. However, the KSA method with both the KKT and value function formulations performs significantly better than the other solution methods where $B \geq 40$. In contrast, we do not find the "step" around $B = 40$ and instead find that the computational time acts quite linearly.

Figure 5.11 shows how the computational times for each formulation as M is varied. As we can see, the computational times remain relatively consistent over all values of M. From the additional plots in [149], we can see that for each value of B, the computational time remains relatively constant for all values of M. However, as each value

of B increases, the constant value also increases. This behaviour implies that the leaders budget constraint has a much larger impact on the computational times than the maximum taxation does.

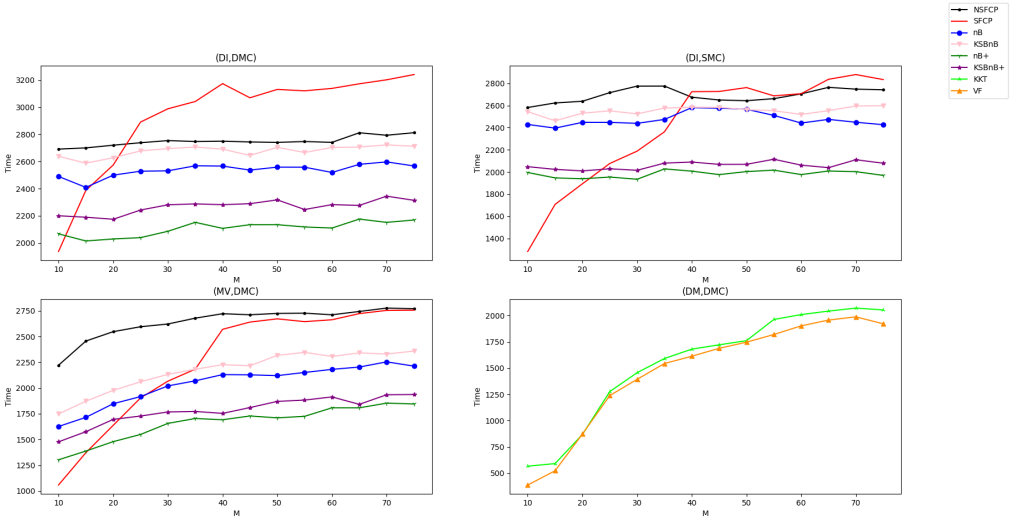


FIGURE 5.11: Set C results as the maximum taxation is varied.

In Figure 5.11 we can see that, once again, with the dichotomic and max value formulations, the nB+ and KSBnB+ solution methods perform best with respect to computational time. It would also appear that for small values of M, the KSA, method with either the KKT or value function formulations, perform quickest. However, when $B \leq 35$, we find that the KSA method is competitive for when $M = 10$ or 15, however, for larger values of M, the nB+ solution method with the max value formulation is considerably quicker. Yet, when $B \geq 40$, the KSA method is consistantly quicker than all other solution methods with both the dichotomic and max value formulations.

5.4 Results Overall

Across all three instances we see that the KSA algorithm performs extremely well compared to the other solution methods presented. Not only did it solve the highest proportion of problems in Set A with the non-unit supply instances, its computational time was drastically smaller with unit supply problems. In Set B, KSA achieved the fastest computational times along with the highest proportion of instances solved and was fastest in Set C for $B \geq 40$ and for $B \leq 35$ when $M = 10$ or 15.

Chapter 6

Conclusions

Bilevel optimisation problems emerge from a vast variety of applications across multiple industries. In this thesis, we have focused on a specific bilevel problem based upon commodities being bought from a market. The follower's objective is to purchase a set of commodities which satisfy some combinatorial constraints in the cheapest manor possible. Whilst the leader aims to generate profit by purchasing commodities from the market and then selling them to the follower at some inflated price. Due to the binary decisions of the follower, the resulting problem can be described as a Mixed Integer Non-Linear Bilevel Optimisation problem. As there already exists state-of-the-art solvers readily available, the aim was to reformulate our MINLBP into a single-level MILP, solved using tailored solution methods which could be carried out by using the solvers built in callbacks.

In Chapter 2 we conducted a literature review discussing properties, applications and solution methods of bilevel problems. To begin with we introduced some basic terminology with regards to the feasible regions of a bilevel problem compared to a standard MILP. Following this, we discussed the applications and focused on Facility Location, Interdiction and Pricing problems as a large proportion of the literature was focused to these special cases. Following this, we discussed some bilevel properties before moving on to existing solution methods. A majority of the solution methods for the non-integer problems focused on reformulating the bilevel problem to a single level using methods such as KKT, value function constraints and reaction set mapping, to name a few. However, for problems containing integer variables, there was a large focus towards using Branch-and-Bound or Branch-and-Cut frameworks with modified fathoming rules along with tailored cutting planes.

In Chapter 3 we introduce the Bilevel Pricing Problem we wish to solve. We begin by focusing on the unit supply case with a single follower and introduced theorems relating to the bound on the leader's objective and the lack of need for negative taxations. Then, after providing the unit supply case with multiple followers, we moved

on to the more general case with multiple followers and a non-unit supply for each commodity. Here we discover that as the followers must decide which commodities to purchase from the leader or the market, we had to introduce a max operator to both the leader and follower objective functions. As most readily available solvers would not be able to handle this max operator, we have to find a way of removing it. Thus, we presented two reformulation methods which we named the dicotomic and max value formulations. In the dicotomic formulation we introduced an additional set of follower variables which indicate the commodities bought from the leader or market. Whereas in the max value formulation we introduce a binary variables which determined which part of the max operation took the greater value.

Up until this point, all of our formulations have been non-linear. To linearise our formulations, we used McCormick envelopes along with binary expansions if necessary. After briefly discussing how [53] could have generated a smaller feasible region using McCormick envelopes and a value function constraint rather than their own cutting plane, which did not require any additional variables, we move onto discussing McCormick envelopes when used within a summation. We discussed how $X = \sum_{j=1}^{n} \sum_{i=1}^{m} y_{ij} x_j$ can be linearised using either a direct McCormick or a summation McCormick because of x being independent of i. We discovered that the DMC and SMC would provide tighter upper and lower bounds, respectively, for all values of m expect when $m = 2^k - 1$ for some $k \in \mathbb{N}$, in which case the bounds would be equal. Likewise, we showed how, if there exists the constraint $\sum_{i=1}^{m} y_{ij} \leq s$, should $s \leq 2^{\lfloor \log_2(m+1) \rfloor} - 1$, then the SMC shall provide an upper bound at least as tight as DMC and visa versa for the lower bound.

In Chapter 4, after providing several formulations for our bilevel problem, we move towards solution methods. For the unit supply case, we noticed that the feasible region of the follower's problem was completely independent of the leader's variables, which meant that we could perform a simple cutting-plane algorithm within a branch-and-cut framework. At every integer solution $(x^*, \bar{x}^*, y^*)$ discovered, we simply solve the follower's problem, parameterised by the corresponding leader's variables, and if we discover a solution which provides a better objective for the follower we introduce the relevant value function cutting plane. As a result of the independence of the follower's feasible region, this cutting-plane was globally valid and could be enforced globally. However, in the more general non-unit supply case, applying the same cutting-plane solution method would provide an incorrect solution.

In the non-unit supply scenario, should the follower wish to purchase a commodity, they have the choice between purchasing from the leader or the market. In such case, given that we have only assumed positive taxation by the leader, the follower shall only purchase commodity j from the leader if the amount they need is greater than the amount the leader has not purchased, i.e., what is left on the market. As a result, the follower's decision now relies on the leader's variables and we have lost the independence between the feasible regions. Thus, the cutting plane method cannot be

used directly as the cutting planes are no longer globally valid. Therefore, we devise four solution methods, two of which use a column generation approach with the other two using a sophisticated branching strategy. The first column generation approach focused on the exact solution that needed to be removed, whereas the latter concentrate on the set of commodities, which should require fewer cutting planes and thus column generations. With the branching strategies, rather than introducing variables to indicate when a constraint should be active, we partition the feasible region into multiple sections, one of which we would apply our cutting plane.

Following this, we discuss solution methods where the leader has already computed all feasible follower solutions and how this information can be used with specific solution methods. Initially we introduced reformulations for when the leader knows of all feasible follower solutions and demonstrated how the resulting formulations have a totally unimodular follower's problem. This allows us to use techniques such as KKT and strong duality, which are usually unavailable for problems which contain integer variables in the lower level. However, computing all of the follower's feasible solutions is unattainable in the majority of cases. Therefore we introduce the KSA algorithm in which we assume the follower's responds with a solution from a subset of feasible points. At every bilevel infeasible point, we would add any new follower points to the subset until we reach a bilevel feasible solution. At which point we shall solve the bilevel problem using one of the previous solution methods, applying a lower bound on the objective function and forcing the follower to select a response outside of the subset.

The downside of the KSA algorithm is that we have to generate and solve two formulations, whereas ideally we want to encompass the whole process within a single framework, which can be done with the KSB method. With the KSB method, whilst using any of the unit or non-unit supply solution methods, should the solver decide to branch from the root node, we would generate a subset Y of feasible follower solutions and generate $|Y| + 1$ child nodes, where we have either fixed the followers response to one from Y or stopped them from using a solution in Y, i.e., $y \notin Y$.

Finally, in Chapter 5 we presented the computational results from three sets of instances. For Set A, we varied the number of commodities along with the supply for each commodity. For the non-unit supply case, we found that the KSA algorithm with the value function formulation performs best with respect to time, with the nB+ solution methods performing best with the dichotomic and max value formulations. Along with this, we found that the double MINLBP method solves the follower's problem far fewer times than the nB+ solution method. Thus, for problems where the follower's problem is much more complicated, the double MINLBP method is much more preferable. For the unit supply case we discovered that the CP solution method with the summation McCormick formulation performs the best across all instances. It is expected that the

unit supply solution methods would outperform the non-unit supply solution methods when applied to the unit supply case. However, the difference between the times are far greater than we predicted, with the CP method with the summation McCormick formulation averaging below 2 seconds for all instances.

For Set B, we varied the number of followers along with the supply of each commodity. Once again, we found that the nB+ solution method performs very well with both dichotomic and max value formulations, with the KSA solution method providing the quickest computational times. Set B was designed to compare the performances of the direct and summation McCormick formulations and see if the computational results match our predictions from Section 3.7.2.1. From our predictions, we expected the summation McCormick to be quicker than the direct formulation when $s_{\text{coef}} < 1$, vice versa for $s_{\text{coef}} > 1$ and both formulations to be the same at $s_{\text{coef}} = 1$. However, we discover that for the NSFCP, SFCP and nB+ solutions methods the summation McCormick formulation is consistently faster than its direct counterpart for all values of s_{coef}. With the remaining solution methods, the summation McCormick was always quicker when $s_{\text{coef}} < 1$ with some instances of the direct being quicker when $s_{\text{coef}} > 1$.

With Set C, we want to observe how the values of M and B affect the performance of the formulations and solution methods. From our results, we discover that B had a much greater impact on the computational times as increasing B can drastically increase the possible combinations of commodities that the leader can purchase. In general, we once again find that the KSA algorithm perform very well overall. However, when $B \leq 35$ we see that the nB+ solution method with the max value formulation actually achieve solutions faster than the KSA method. This implies that for the "harder" problems, with a higher leader budget the KSA method should be chosen, however with the "easier" smaller budget problems, it is advantageous to contain everything within a single Branch-and-Cut framework.

There are a number of avenues to be explored relating to this thesis. Firstly, from our computational results we found that the KSA method performs very well, with the first MINLBP attaining the optimal bilevel solution in a high proportion of instances. The initial set Y was just computed as the $|Y|$-many cheapest solutions for the follower when there is zero taxation applied. By developing a more sophisticated strategy for generating Y, using Section 4.3.2.1 as a starting point, we may be able to show that only the first MINLBP is needed. This can also be used with the Known Solution Branching method from Section 4.3.3 to improve the KSBnB and KSBnB+ solution methods. Secondly, we can explore the summation McCormicks even further. In our formulations and computational results we focus our attention to the *logarithmic* reformulation. However, as discussed in Section 3.7, there exists two additional binarisation techniques. Ideally, we would perform the same computational results from Sets A, B and C, with *full* and *unary* to determine which summation McCormick method would

have the best performance. Lastly, in Examples 3.2 and 3.3 we show how when the follower's respond in an order, and do not act collectively, then the optimal solution is not the same. Thus, in any future works we want to explore this case, to see if the solution methods presented can be directly applied and if so do they perform similarly to our instances. Likewise, we should also explore formulations and solution methods for the negative taxation setting, given that Example 3.3 demonstrates how this can increase the objective value for the leader.

Appendix A

Linear Formulations

A.1 Unit Supply

A.1.1 Unit Supply Formulation

A.1.1.1 Direct McCormick

$$\max_{x,\bar{x}} \quad \sum_{j\in\mathcal{J}} \Big(\sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} (p_j^{ik} + q_j^{ik} c_j) - \bar{x}_j c_j \Big) \tag{A.1a}$$

$$\text{s.t.} \quad x_j \leq M\bar{x}_j \quad \forall j \in \mathcal{J}, \tag{A.1b}$$

$$\sum_{j\in\mathcal{J}} \bar{x}_j c_j \leq B, \tag{A.1c}$$

$$\mathcal{MC}(p_j^{ik}, y_j^{ik}, x_j) \quad \forall j \in \mathcal{J}, i \in \mathcal{I}, k \in \{1,\dots,d^i\} \tag{A.1d}$$

$$\mathcal{MC}(q_j^{ik}, y_j^{ik}, \bar{x}_j) \quad \forall j \in \mathcal{J}, i \in \mathcal{I}, k \in \{1,\dots,d^i\}, \tag{A.1e}$$

$$x \in [0, M]^n, \tag{A.1f}$$

$$\bar{x} \in \{0,1\}^n, \tag{A.1g}$$

$$p^{ik} \in [0, M]^n \quad \forall i \in \mathcal{I}, k \in \{1,\dots,d^i\}, \tag{A.1h}$$

$$q^{ik} \in \{0,1\}^n \quad \forall i \in \mathcal{I}, k \in \{1,\dots,d^i\}, \tag{A.1i}$$

$$y \in \arg\min_{y} \{ \sum_{j\in\mathcal{J}} \sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} (p_j^{ik} + y_j^{ik} c_j) : \tag{A.1j}$$

$$A^i y_j^{ik} \leq b^i \quad \forall i \in \mathcal{I}, k \in \{0,\dots,d^i\}, \tag{A.1k}$$

$$\sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} y_j^{ik} \leq 1 \quad \forall j \in \mathcal{J}, \tag{A.1l}$$

$$y^{ik} \in \{0,1\}^n \quad \forall i \in \mathcal{I}, k \in \{1,\dots,d^i\}\}. \tag{A.1m}$$

A.1.1.2 Summation McCormick

$$\max_{x,\bar{x}} \quad \sum_{j\in\mathcal{J}} \Big(p_j + q_j c_j - \bar{x}_j c_j \Big) \tag{A.2a}$$

$$\text{s.t.} \quad x_j \leq M\bar{x}_j \quad \forall j \in \mathcal{J}, \tag{A.2b}$$

$$\sum_{j\in\mathcal{J}} \bar{x}_j c_j \leq B, \tag{A.2c}$$

$$\mathcal{MC}(p_j, \sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} y_j^{ik}, x_j) \quad \forall j \in \mathcal{J}, \tag{A.2d}$$

$$\mathcal{MC}(q_j, \sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} y_j^{ik}, \bar{x}_j) \quad \forall j \in \mathcal{J}, \tag{A.2e}$$

$$x \in [0, M]^n, \tag{A.2f}$$

$$\bar{x} \in \{0,1\}^n, \tag{A.2g}$$

$$p \in [0, M]^n \quad \forall j \in \mathcal{J}, \tag{A.2h}$$

$$q \in \{0,1\}^n \quad \forall j \in \mathcal{J}, \tag{A.2i}$$

$$y \in \arg\min_{y} \{ \sum_{j\in\mathcal{J}} \sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} (p_j^{ik} + y_j^{ik} c_j) : \tag{A.2j}$$

$$A^i y_j^{ik} \leq b^i \quad \forall i \in \mathcal{I}, k \in \{0, \ldots, d^i\}, \tag{A.2k}$$

$$\sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} y_j^{ik} \leq 1 \quad \forall j \in \mathcal{J}, \tag{A.2l}$$

$$y^{ik} \in \{0,1\}^n \quad \forall i \in \mathcal{I}, k \in \{1, \ldots, d^i\}\}. \tag{A.2m}$$

A.1.2 Known Solution Formulations

A.1.2.1 KKT

$$\max_{x,\bar{x}} \quad \sum_{l=1}^{|\mathcal{Y}^{Sol}|} \Big(\sum_{j\in\mathcal{J}} \sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} y_{l_j^{ik}} (p_{lj} + q_{lj} c_j) \Big) - \sum_{j\in\mathcal{J}} \bar{x}_j c_j \tag{A.3a}$$

$$\text{s.t.} \quad x_j \leq M\bar{x}_j \quad \forall j \in \mathcal{J}, \tag{A.3b}$$

$$\sum_{j\in\mathcal{J}} \bar{x}_j c_j \leq B, \tag{A.3c}$$

$$\mathcal{MC}(p_{lj}, \lambda_l, x_j) \quad \forall l \in \{1,\dots,|\mathcal{Y}^{Sol}|\}, \forall j \in \mathcal{J}, \tag{A.3d}$$

$$\mathcal{MC}(q_{lj}, \lambda_l, \bar{x}_j) \quad \forall l \in \{1,\dots,|\mathcal{Y}^{Sol}|\}, \forall j \in \mathcal{J}, \tag{A.3e}$$

$$0 = \sum_{j\in\mathcal{J}} \sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} y_{l_j^{ik}} (x_j + c_j) - \mu_l + \nu \quad \forall l \in \{1,\dots,|\mathcal{Y}^{Sol}|\}, \tag{A.3f}$$

$$\text{SOS Type 1}(\mu_l, \lambda_l) \quad \forall l \in \{1,\dots,|\mathcal{Y}^{Sol}|\}, \tag{A.3g}$$

$$0 = \lambda_1 + \dots + \lambda_{|\mathcal{Y}^{Sol}|} - 1, \tag{A.3h}$$

$$x \in [0, M]^n, \tag{A.3i}$$

$$\bar{x} \in \{0, 1\}^n, \tag{A.3j}$$

$$\lambda, \mu \geq 0, \tag{A.3k}$$

$$\nu \, free, \tag{A.3l}$$

$$p_l \in [0, M]^n \quad \forall l \in \{1,\dots,|\mathcal{Y}^{Sol}|\}, \tag{A.3m}$$

$$q_l \in [0, s_j]^n \quad \forall l \in \{1,\dots,|\mathcal{Y}^{Sol}|\}. \tag{A.3n}$$

A.1.2.2 Value Function

$$\max_{x,\bar{x}} \quad \sum_{l=1}^{|\mathcal{Y}^{Sol}|} \sum_{j\in\mathcal{J}} \sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} y_{l_j}^{ik}(p_{lj} + q_{lj}c_j) - \sum_{j\in\mathcal{J}} \bar{x}_j c_j \tag{A.4a}$$

$$\text{s.t.} \quad x_j \leq M\bar{x}_j \quad \forall j \in \mathcal{J}, \tag{A.4b}$$

$$\sum_{j\in\mathcal{J}} \bar{x}_j c_j \leq B, \tag{A.4c}$$

$$\mathcal{MC}(p_{lj}, \lambda_l, x_j) \quad \forall l \in \{1,\dots,|\mathcal{Y}^{Sol}|\}, \forall j \in \mathcal{J}, \tag{A.4d}$$

$$\mathcal{MC}(q_{lj}, \lambda_l, \bar{x}_j) \quad \forall l \in \{1,\dots,|\mathcal{Y}^{Sol}|\}, \forall j \in \mathcal{J}, \tag{A.4e}$$

$$\sum_{l=1}^{|\mathcal{Y}^{Sol}|} \sum_{j\in\mathcal{J}} \sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} y_{l_j}^{ik}(p_{lj} + \lambda_l c_j) \leq \sum_{j\in\mathcal{J}} \sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} y_{o_j}^{ik}(x_j + c_j) \quad \forall o \in \{1,\dots,|\mathcal{Y}^{Sol}|\}, \tag{A.4f}$$

$$0 = \lambda_1 + \cdots + \lambda_{|\mathcal{Y}^{Sol}|} - 1, \tag{A.4g}$$

$$x \in [0, M]^n, \tag{A.4h}$$

$$\bar{x} \in \{0, 1\}^n, \tag{A.4i}$$

$$p_l \in [0, M]^n \quad \forall l \in \{1,\dots,|\mathcal{Y}^{Sol}|\}, \tag{A.4j}$$

$$q_l \in [0, s_j]^n \quad \forall l \in \{1,\dots,|\mathcal{Y}^{Sol}|\}. \tag{A.4k}$$

A.2 Non-Unit Supply

A.2.1 Dichotomic Formulation

A.2.1.1 Direct McCormick

$$\max_{x,\bar{x}} \quad \sum_{j\in\mathcal{J}} \Big(\sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} p_j^{ik} + \bar{y}_j^{ik} c_j \Big) - \bar{x}_j c_j \tag{A.5a}$$

$$\text{s.t.} \quad x_j \le M\bar{x}_j \quad \forall j \in \mathcal{J}, \tag{A.5b}$$

$$\sum_{j\in\mathcal{J}} \bar{x}_j c_j \le B, \tag{A.5c}$$

$$\bar{x}_j \le s_j \quad \forall j \in \mathcal{J}, \tag{A.5d}$$

$$\mathcal{MC}(p_j^{ik}, \bar{y}_j^{ik}, x_j) \quad \forall j \in \mathcal{J}, \forall i \in \mathcal{I}, \forall k \in \{1,\dots,d^i\}, \tag{A.5e}$$

$$p_j^{ik} \in [0, M] \quad \forall j \in \mathcal{J}, \forall i \in \mathcal{I}, \forall k \in \{1,\dots,d^i\}, \tag{A.5f}$$

$$x \in [0, M]^n, \tag{A.5g}$$

$$\bar{x} \in \mathbb{Z}^n_{\ge 0}, \tag{A.5h}$$

$$y, \bar{y} \in \arg\min_{y,\bar{y}} \{ \sum_{j\in\mathcal{J}} \sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} p_j^{ik} + \bar{y}_j^{ik} c_j + y_j^{ik} c_j : \tag{A.5i}$$

$$A^i(\bar{y}^{ik} + y^{ik}) \le b^i \quad \forall i \in \mathcal{I}, \forall k \in \{1,\dots,d^i\}, \tag{A.5j}$$

$$\sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} \bar{y}_j^{ik} + y_j^{ik} \le s_j \quad \forall j \in \mathcal{J}, \tag{A.5k}$$

$$\sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} \bar{y}_j^{ik} \le \bar{x}_j \quad \forall j \in \mathcal{J}, \tag{A.5l}$$

$$\sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} y_j^{ik} \le s_j - \bar{x}_j \quad \forall j \in \mathcal{J}, \tag{A.5m}$$

$$\bar{y}_j^{ik} + y_j^{ik} \le 1 \quad \forall j \in \mathcal{J}, \forall i \in \mathcal{I}, \forall k \in \{1,\dots,d^i\}, \tag{A.5n}$$

$$\bar{y}^{ik}, y^{ik} \in \{0,1\}^n \}. \tag{A.5o}$$

A.2.1.2 Summation McCormick

$$\max_{x,\bar{x}} \quad \sum_{j\in\mathcal{J}} \Big(\sum_{l=1}^{\lfloor \log_2 s_j \rfloor + 1} 2^{l-1} p_{jl} + \sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} \bar{y}_j^{ik} c_j - \bar{x}_j c_j \Big) \tag{A.6a}$$

$$\text{s.t.} \quad x_j \le M \bar{x}_j \quad \forall j \in \mathcal{J}, \tag{A.6b}$$

$$\sum_{j\in\mathcal{J}} \bar{x}_j c_j \le B, \tag{A.6c}$$

$$\bar{x}_j \le s_j \quad \forall j \in \mathcal{J}, \tag{A.6d}$$

$$\sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} \bar{y}_j^{ik} = \sum_{l=1}^{\lfloor \log_2 s_j \rfloor + 1} 2^{l-1} \alpha_{jl} \quad \forall j \in \mathcal{J}, \tag{A.6e}$$

$$\mathcal{MC}(p_{jl}, \alpha_{jl}, x_j) \quad \forall j \in \mathcal{J}, \forall l \in \{1, \dots, \lfloor \log_2 s_j \rfloor + 1\}, \tag{A.6f}$$

$$\alpha_{jl} \in \{0,1\} \quad \forall j \in \mathcal{J}, \forall l \in \{1, \dots, \lfloor \log_2 s_j \rfloor + 1\}, \tag{A.6g}$$

$$p_{jl} \in [0, M] \quad \forall j \in \mathcal{J}, \forall l \in \{1, \dots, \lfloor \log_2 s_j \rfloor + 1\}, \tag{A.6h}$$

$$x \in [0, M]^n, \tag{A.6i}$$

$$\bar{x} \in \mathbb{Z}^n_{\ge 0}, \tag{A.6j}$$

$$y, \bar{y} \in \arg\min_{y,\bar{y}} \{ \sum_{j\in\mathcal{J}} \Big(\sum_{l=1}^{\lfloor \log_2 s_j \rfloor + 1} 2^{l-1} p_{jl} + \sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} \bar{y}_j^{ik} c_j + y_j^{ik} c_j \Big) : \tag{A.6k}$$

$$A^i(\bar{y}^{ik} + y^{ik}) \le b^i \quad \forall i \in \mathcal{I}, \forall k \in \{1, \dots, d^i\}, \tag{A.6l}$$

$$\sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} \bar{y}_j^{ik} + y_j^{ik} \le s_j \quad \forall j \in \mathcal{J}, \tag{A.6m}$$

$$\sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} \bar{y}_j^{ik} \le \bar{x}_j \quad \forall j \in \mathcal{J}, \tag{A.6n}$$

$$\sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} y_j^{ik} \le s_j - \bar{x}_j \quad \forall j \in \mathcal{J}, \tag{A.6o}$$

$$\bar{y}_j^{ik} + y_j^{ik} \le 1 \quad \forall j \in \mathcal{J}, \forall i \in \mathcal{I}, \forall k \in \{1, \dots, d^i\}, \tag{A.6p}$$

$$\bar{y}^{ik}, y^{ik} \in \{0,1\}^n \}. \tag{A.6q}$$

A.2.2 Max Value Formulation

$$\max_{x,\bar{x},w,\alpha,\beta} \quad \sum_{j\in\mathcal{J}} \Big(\sum_{l=1}^{\lfloor \log_2 s_j \rfloor + 1} 2^{l-1} p_{jl} \Big) + z_j c_j - \bar{x}_j c_j \tag{A.7a}$$

$$\text{s.t.} \quad x_j \le M\bar{x}_j \quad \forall j \in \mathcal{J}, \tag{A.7b}$$

$$\sum_{j\in\mathcal{J}} \bar{x}_j c_j \le B, \tag{A.7c}$$

$$\bar{x}_j \le s_j \quad \forall j \in \mathcal{J}, \tag{A.7d}$$

$$z_j = \sum_{i=1}^{\lfloor \log_2 s_j \rfloor + 1} 2^{l-1} \alpha_{jl} \quad \forall j \in \mathcal{J}, \tag{A.7e}$$

$$\mathcal{MC}(p_{lj}, \alpha_{jl}, x_j) \quad \forall j \in \mathcal{J}, \forall l \in \{1, \dots, \lfloor \log_2 s_j \rfloor + 1\}, \tag{A.7f}$$

$$\alpha_{jl} \in \{0,1\} \quad \forall j \in \mathcal{J}, \forall l \in \{1, \dots, \lfloor \log_2 s_j \rfloor + 1\}, \tag{A.7g}$$

$$p_{jl} \in [0, M] \quad \forall j \in \mathcal{J}, \forall l \in \{1, \dots, \lfloor \log_2 s_j \rfloor + 1\}, \tag{A.7h}$$

$$x \in [0, M]^n, \tag{A.7i}$$

$$\bar{x} \in \mathbb{Z}^n_{\ge 0}, \tag{A.7j}$$

$$y, \gamma, z \in \arg\min_{y,\gamma,z} \{ \sum_{j\in\mathcal{J}} \Big(\sum_{l=1}^{\lfloor \log_2 s_j \rfloor + 1} 2^{l-1} p_{jl} \Big) + \sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} y_j^{ik} c_j : \tag{A.7k}$$

$$A^i y^{ik} \le b^i \quad \forall i \in \mathcal{I}, \forall k \in \{1, \dots, d^i\}, \tag{A.7l}$$

$$\sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} y_j^{ik} \le s_j \quad \forall j \in \mathcal{J}, \tag{A.7m}$$

$$\bar{M}(1-\gamma_j) \ge -\Big(\sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} y_j^{ik} - (s_j - \bar{x}_j) \Big) \quad \forall j \in \mathcal{J}, \tag{A.7n}$$

$$\bar{M}\gamma_j \ge \sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} y_j^{ik} - (s_j - \bar{x}_j) \quad \forall j \in \mathcal{J}, \tag{A.7o}$$

$$z_j \ge 0 \quad \forall j \in \mathcal{J}, \tag{A.7p}$$

$$z_j \ge \sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} y_j^{ik} - (s_j - \bar{x}_j) \quad \forall j \in \mathcal{J}, \tag{A.7q}$$

$$z_j \le s_j \gamma_j \quad \forall j \in \mathcal{J}, \tag{A.7r}$$

$$z_j \le \sum_{i\in\mathcal{I}} \sum_{k=1}^{d^i} y_j^{ik} - (s_j - \bar{x}_j) + s_j(1-\gamma_j) \quad \forall j \in \mathcal{J}, \tag{A.7s}$$

$$\gamma \in \{0,1\}^n, \tag{A.7t}$$

$$z \in \mathbb{Z}^n_{\ge 0}\}. \tag{A.7u}$$

A.2.3 Double MINLBP Formulation

A.2.3.1 KKT

$$\max_{x,\bar{x},\lambda,\gamma,z,\mu,\nu,\alpha,p,q} \quad \sum_{l=1}^{|\mathcal{Y}^{Sol}|}\Big(\sum_{j\in\mathcal{J}}\sum_{m=1}^{\lfloor\log_2 z_{lj}\rfloor} 2^{m-1}q_{ljm} + r_{lj}c_j\Big) - \sum_{j\in\mathcal{J}}\bar{x}_j c_j \tag{A.8a}$$

$$\text{s.t.} \quad x_j \le M\bar{x}_j \quad \forall j \in \mathcal{J}, \tag{A.8b}$$

$$\sum_{j\in\mathcal{J}}\bar{x}_j c_j \le B, \tag{A.8c}$$

$$\bar{x}_j \le s_j \quad \forall j \in \mathcal{J}, \tag{A.8d}$$

$$\bar{M}(1-\gamma_{lj}) \ge -\Big(\sum_{i\in\mathcal{I}}\sum_{k=1}^{d^i} y_{l_j}^{ik} - (s_j - \bar{x}_j)\Big) \quad \forall j \in \mathcal{J}, \tag{A.8e}$$

$$\bar{M}\gamma_{lj} \ge \sum_{i\in\mathcal{I}}\sum_{k=1}^{d^i} y_{l_j}^{ik} - (s_j - \bar{x}_j) \quad \forall j \in \mathcal{J}, \tag{A.8f}$$

$$\mathcal{MC}(z_{lj}, \sum_{i\in\mathcal{I}}\sum_{k=1}^{d^i} y_{l_j}^{ik} - (s_j - \bar{x}_j), \gamma_{lj}) \quad \forall l \in \{1,\dots,|\mathcal{Y}^{Sol}|\},\ \forall j \in \mathcal{J}, \tag{A.8g}$$

$$z_{lj} = \sum_{m=1}^{\lfloor\log_2 z_{lj}\rfloor+1} 2^{m-1}\alpha_{ljm} \quad \forall l \in \{1,\dots,|\mathcal{Y}^{Sol}|\},\ \forall j \in \mathcal{J}, \tag{A.8h}$$

$$\mathcal{MC}(p_{ljm}, \alpha_{ljm}, x_j) \quad \forall l \in \{1,\dots,|\mathcal{Y}^{Sol}|\},\ \forall j \in \mathcal{J},\ \forall m \in \{1,\dots,\lfloor\log_2 z_{lj}\rfloor+1\}, \tag{A.8i}$$

$$\mathcal{MC}(r_{lj}, \lambda_l, z_{lj}) \quad \forall l \in \{1,\dots,|\mathcal{Y}^{Sol}|\},\ \forall j \in \mathcal{J}, \tag{A.8j}$$

$$\mathcal{MC}(q_{ljm}, \lambda_l, p_{ljm}) \quad \forall l \in \{1,\dots,|\mathcal{Y}^{Sol}|\},\ \forall j \in \mathcal{J},\ \forall m \in \{1,\dots,\lfloor\log_2 z_{lj}\rfloor+1\}, \tag{A.8k}$$

$$0 = \sum_{j\in\mathcal{J}}\Big(\sum_{m=1}^{\lfloor\log_2 z_{lj}\rfloor+1} 2^{m-1}p_{ljm}\Big) + \sum_{i\in\mathcal{I}}\sum_{k=1}^{d^i} y_{l_j}^{ik}c_j) - \mu_l + \nu \quad \forall l \in \{1,\dots,|\mathcal{Y}^{Sol}|\}, \tag{A.8l}$$

$$\text{SOS Type 1}(\mu_l, \lambda_l) \quad \forall l \in \{1,\dots,|\mathcal{Y}^{Sol}|\}, \tag{A.8m}$$

$$0 = \lambda_1 + \cdots + \lambda_{|\mathcal{Y}^{Sol}|} - 1, \tag{A.8n}$$

$$x \in [0, M]^n, \tag{A.8o}$$

$$\bar{x} \in \mathbb{Z}^n_{\ge 0}, \tag{A.8p}$$

$$\lambda, \mu \ge 0, \tag{A.8q}$$

$$\gamma_l \in \{0,1\}^n \quad \forall l \in \{1,\dots,|\mathcal{Y}^{Sol}|\}, \tag{A.8r}$$

$$z_l \in \mathbb{Z}^n_{\ge 0} \quad \forall l \in \{1,\dots,|\mathcal{Y}^{Sol}|\}, \tag{A.8s}$$

$$\nu \text{ free}, \tag{A.8t}$$

$$p_{lj}, q_{lj} \in [0, M]^{\lfloor\log_2 z_{lj}\rfloor+1} \quad \forall l \in \{1,\dots,|\mathcal{Y}^{Sol}|\}, j \in \mathcal{J}, \tag{A.8u}$$

$$r_l \in [0, M]^n \quad \forall l \in \{1,\dots,|\mathcal{Y}^{Sol}|\}. \tag{A.8v}$$

A.2.3.2 Value Function

$$\max_{x,\bar{x},\lambda,z,\gamma,\alpha,p,q,r} \quad \sum_{l=1}^{|\mathcal{Y}^{Sol}|}\Big(\sum_{j\in\mathcal{J}}\sum_{m=1}^{\lfloor\log_2 z_{lj}\rfloor} 2^{m-1}q_{ljm} + r_{lj}c_j\Big) - \sum_{j\in\mathcal{J}}\bar{x}_j c_j \tag{A.9a}$$

$$\text{s.t.} \quad (3.4b) - -(3.4f), \tag{A.9b}$$

$$(4.17) - (4.18) \quad \forall l \in \{1,\dots,|\mathcal{Y}^{Sol}|\},\ \forall j \in \mathcal{J}, \tag{A.9c}$$

$$z_{lj} = \sum_{m=1}^{\lfloor\log_2 z_{lj}\rfloor} 2^{m-1}\alpha_{ljm} \quad \forall l \in \{1,\dots,|\mathcal{Y}^{Sol}|\},\ \forall j \in \mathcal{J}, \tag{A.9d}$$

$$\mathcal{MC}(p_{ljm}, \alpha_{ljm}, x_j) \quad \forall l \in \{1,\dots,|\mathcal{Y}^{Sol}|\},\ \forall j \in \mathcal{J},\ \forall m \in \{1,\dots,\lfloor\log_2 z_{lj}\rfloor\}, \tag{A.9e}$$

$$\mathcal{MC}(r_{lj}, \lambda_l, z_{lj}) \quad \forall l \in \{1,\dots,|\mathcal{Y}^{Sol}|\},\ \forall j \in \mathcal{J}, \tag{A.9f}$$

$$\mathcal{MC}(q_{ljm}, \lambda_l, p_{ljm}) \quad \forall l \in \{1,\dots,|\mathcal{Y}^{Sol}|\},\ \forall j \in \mathcal{J},\ \forall m \in \{1,\dots,\lfloor\log_2 z_{lj}\rfloor\}, \tag{A.9g}$$

$$\sum_{l=1}^{\mathcal{Y}^{Sol}} \lambda_l = 1, \tag{A.9h}$$

$$\sum_{l=1}^{|\mathcal{Y}^{Sol}|}\sum_{j\in\mathcal{J}}\sum_{m=1}^{\lfloor\log_2 z_{lj}\rfloor}\Big(2^{m-1}q_{ljm}\Big) + \lambda_l\Big(\sum_{i\in\mathcal{I}}\sum_{k=1}^{d^i} y_l{}_j^{ik}\Big)c_j \leq,$$

$$\sum_{j\in\mathcal{J}}\sum_{m=1}^{\lfloor\log_2 z_{lj}\rfloor}\Big(2^{m-1}p_{ojm}\Big) + \sum_{i\in\mathcal{I}}\sum_{k=1}^{d^i} y_o{}_j^{ik}c_j \quad \forall o \in \{1,\dots,|\mathcal{Y}^{Sol}|\}, \tag{A.9i}$$

$$\lambda \in \{0,1\}^{|\mathcal{Y}^{Sol}|}, \tag{A.9j}$$

$$p_{lj}, q_{lj} \in [0,M]^{\lfloor\log_2 z_{lj}\rfloor+1} \quad \forall l \in \{1,\dots,|\mathcal{Y}^{Sol}|\}, j \in \mathcal{J}, \tag{A.9k}$$

$$r_l \in [0,M]^n. \tag{A.9l}$$

References

[1] MS Windows NT kernel description. https://www.futbin.com/squad-building-challenges/ALL/2482/Adebayo%20Akinfenwa. Accessed: 11-05-2022.

[2] A. Akbari-Dibavar, B. Mohammadi-Ivatloo, and K. Zare. Optimal stochastic bilevel scheduling of pumped hydro storage systems in a pay-as-bid energy market environment. *Journal of Energy Storage*, 31:101608, 2020.

[3] D. Aksen and N. Aras. A matheuristic for leader-follower games involving facility location-protection-interdiction decisions. In *Metaheuristics for Bi-level Optimization*, pages 115–151. Springer, 2013.

[4] F. A. Al-Khayyal and J. E. Falk. Jointly constrained biconvex programming. *Mathematics of Operations Research*, 8(2):273–286, 1983.

[5] G. Anandalingam and D. White. A solution method for the linear static stackelberg problem using penalty functions. *IEEE Transactions on Automatic Control*, 35(10):1170–1173, 1990.

[6] G. Angulo and M. Van Vyve. Fixed-charge transportation problems on trees. *Operations Research Letters*, 45(3):275–281, 2017.

[7] J. M. Arroyo and F. J. Fernández. A genetic algorithm approach for the analysis of electric grid interdiction with line switching. In *2009 15th International Conference on Intelligent System Applications to Power Systems*, pages 1–6. IEEE, 2009.

[8] A. Aswani, Z.-J. Shen, and A. Siddiq. Inverse optimization with noisy data. *Operations Research*, 66(3):870–892, 2018.

[9] S. Avraamidou and E. N. Pistikopoulos. A multi-parametric optimization approach for bilevel mixed-integer linear and quadratic programming problems. *Computers & Chemical Engineering*, 125:98–113, 2019.

[10] J. Bard and B. Golany. Preface-book reviews-practical bilevel optimization: Algorithms and applications. *IIE Transactions*, 31(9):921, 1999.

[11] J. F. Bard. An efficient point algorithm for a linear two-stage optimization problem. *Operations Research*, 31(4):670–684, 1983.

[12] J. F. Bard. An investigation of the linear three level programming problem. *IEEE Transactions on Systems, Man, and Cybernetics*, (5):711–717, 1984.

[13] J. F. Bard. Optimality conditions for the bilevel programming problem. *Naval research logistics quarterly*, 31(1):13–26, 1984.

[14] J. F. Bard. Some properties of the bilevel programming problem. *Journal of optimization theory and applications*, 68(2):371–378, 1991.

[15] J. F. Bard and J. T. Moore. A branch and bound algorithm for the bilevel programming problem. *SIAM Journal on Scientific and Statistical Computing*, 11(2):281–292, 1990.

[16] J. F. Bard and J. T. Moore. An algorithm for the discrete bilevel programming problem. *Naval Research Logistics (NRL)*, 39(3):419–435, 1992.

[17] N. Basilico, S. Coniglio, N. Gatti, and A. Marchesi. Bilevel programming approaches to the computation of optimistic and pessimistic single-leader-multi-follower equilibria. In *SEA*, volume 75, pages 1–14. Schloss Dagstuhl-Leibniz-Zentrum fur Informatik GmbH, Dagstuhl Publishing, 2017.

[18] M. S. Bazaraa, H. D. Sherali, and C. M. Shetty. *Nonlinear programming: theory and algorithms*. John Wiley & Sons, 2013.

[19] R. Bergmann and R. Herzog. Intrinsic formulation of kkt conditions and constraint qualifications on smooth manifolds. *SIAM Journal on Optimization*, 29(4):2423–2444, 2019.

[20] Z. Bi, P. Calamai, and A. Conn. An exact penalty function approach for the linear bilevel programming problem. *Dept. Syst. Design Eng., Univ. Waterlo, Waterloo, ON, Canada, Rep*, 1989.

[21] W. Bialas and M. Karwan. On two-level optimization. *IEEE transactions on automatic control*, 27(1):211–214, 1982.

[22] W. F. Bialas and M. H. Karwan. Two-level linear programming. *Management science*, 30(8):1004–1020, 1984.

[23] J. W. Blankenship and J. E. Falk. Infinitely constrained optimization problems. *Journal of Optimization Theory and Applications*, 19(2):261–281, 1976.

[24] J. BnnoBRs. Partitioning procedures for solving mixed-variables programming problems '. 1962.

[25] P. Bonami and F. Margot. Cut generation through binarization. In *International Conference on Integer Programming and Combinatorial Optimization*, pages 174–185. Springer, 2014.

[26] J. Bracken and J. T. McGill. Mathematical programs with optimization problems in the constraints. *Operations Research*, 21(1):37–44, 1973.

[27] P. Briest. Uniform budgets and the envy-free pricing problem. In *International Colloquium on Automata, Languages, and Programming*, pages 808–819. Springer, 2008.

[28] L. Brotcorne, M. Labbé, P. Marcotte, and G. Savard. Joint design and pricing on a network. *Operations research*, 56(5):1104–1115, 2008.

[29] G. G. Brown, W. M. Carlyle, R. C. Harney, E. M. Skroch, and R. K. Wood. Interdicting a nuclear-weapons project. *Operations Research*, 57(4):866–877, 2009.

[30] H. I. Calvete, C. Domínguez, C. Galé, M. Labbé, and A. Marin. The rank pricing problem: models and branch-and-cut algorithms. *Computers & Operations Research*, 105:12–31, 2019.

[31] H. I. Calvete, C. Gale, and P. M. Mateo. A new approach for solving linear bilevel problems using genetic algorithms. *European Journal of Operational Research*, 188(1):14–28, 2008.

[32] J.-F. Camacho-Vallejo, Á. E. Cordero-Franco, and R. G. González-Ramírez. Solving the bilevel facility location problem under preferences by a stackelberg-evolutionary algorithm. *Mathematical Problems in Engineering*, 2014, 2014.

[33] W. Candler and R. Townsley. A linear two-level programming problem. *Computers & Operations Research*, 9(1):59–76, 1982.

[34] L. CáNovas, S. GarcíA, M. Labbé, and A. MaríN. A strengthened formulation for the simple plant location problem with order. *Operations Research Letters*, 35(2):141–150, 2007.

[35] M. Caramia and R. Mari. Enhanced exact algorithms for discrete bilevel linear problems. *Optimization Letters*, 9(7):1447–1468, 2015.

[36] M. Caramia and R. Mari. A decomposition approach to solve a bilevel capacitated facility location problem with equity constraints. *Optimization Letters*, 10(5):997–1019, 2016.

[37] A. Chaabani, S. Bechikh, and L. B. Said. A new co-evolutionary decomposition-based algorithm for bi-level combinatorial optimization. *Applied Intelligence*, 48(9):2847–2872, 2018.

[38] Y. Chen and M. Florian. On the geometric structure of linear bilevel programs: a dual approach. *Centre De Recherche Sur Les Transports Publication*, (867), 1992.

[39] D. C. Cho, E. L. Johnson, M. Padberg, and M. Rao. On the uncapacitated plant location problem. i: valid inequalities and facets. *Mathematics of Operations Research*, 8(4):579–589, 1983.

[40] D. C. Cho, M. W. Padberg, and M. Rao. On the uncapacitated plant location problem. ii: facets and lifting theorems. *Mathematics of Operations Research*, 8(4):590–612, 1983.

[41] F. Clarke. Optimization and non-smooth analysis, classics in applied mathematics, vol. 5, society for industrial and applied mathematics (siam), philadelphia, pa, 1990. *J. Convex Anal*, 2(1-2):117–144, 1990.

[42] B. Colson, P. Marcotte, and G. Savard. Bilevel programming: A survey. *4or*, 3(2):87–107, 2005.

[43] S. Coniglio, N. Gatti, and A. Marchesi. Computing a pessimistic stackelberg equilibrium with multiple followers: The mixed-pure case. *Algorithmica*, 82(5):1189–1238, 2020.

[44] S. Coniglio, M. Sirvent, and M. Weibelzahl. Airport capacity extension, fleet investment, and optimal aircraft scheduling in a multilevel market model: quantifying the costs of imperfect markets. *OR Spectrum*, 43(2):367–408, 2021.

[45] H. Cui, F. Li, X. Fang, H. Chen, and H. Wang. Bilevel arbitrage potential evaluation for grid-scale energy storage considering wind power and lmp smoothing effect. *IEEE Transactions on Sustainable Energy*, 9(2):707–718, 2017.

[46] S. Dash, O. Günlük, and R. Hildebrand. Binary extended formulations of polyhedral mixed-integer sets. *Mathematical Programming*, 170(1):207–236, 2018.

[47] M. S. Daskin. What you should know about location modeling. *Naval Research Logistics (NRL)*, 55(4):283–294, 2008.

[48] S. Dempe. *Discrete bilevel optimization problems*. Inst. für Wirtschaftsinformatik, 1996.

[49] S. Dempe. Annotated bibliography on bilevel programming and mathematical programs with equilibrium constraints. 2003.

[50] S. Dempe. *Bilevel optimization: theory, algorithms and applications*. TU Bergakademie Freiberg, Fakultät für Mathematik und Informatik, 2018.

[51] S. Dempe and J. Dutta. Is bilevel programming a special case of a mathematical program with complementarity constraints? *Mathematical programming*, 131(1):37–48, 2012.

[52] S. Dempe and S. Franke. Solution algorithm for an optimistic linear stackelberg problem. *Computers & Operations Research*, 41:277–281, 2014.

[53] S. Dempe and F. M. Kue. Solving discrete linear bilevel optimization problems using the optimal value reformulation. *Journal of Global Optimization*, 68(2):255–277, 2017.

[54] S. Dempe, F. Mefo Kue, and P. Mehlitz. Optimality conditions for mixed discrete bilevel optimization problems. *Optimization*, 67(6):737–756, 2018.

[55] S. Dempe and A. B. Zemkoho. Bilevel road pricing: theoretical analysis and optimality conditions. *Annals of Operations Research*, 196(1):223–240, 2012.

[56] S. Dempe and A. B. Zemkoho. The bilevel programming problem: reformulations, constraint qualifications and optimality conditions. *Mathematical Programming*, 138(1-2):447–473, 2013.

[57] S. DeNegre. *Interdiction and discrete bilevel linear programming*. Lehigh University PhD, 2011.

[58] S. T. DeNegre and T. K. Ralphs. A branch-and-cut algorithm for integer bilevel linear programs. In *Operations research and cyber-infrastructure*, pages 65–78. Springer, 2009.

[59] S. Dewez, M. Labbé, P. Marcotte, and G. Savard. New formulations and valid inequalities for a bilevel pricing problem. *Operations research letters*, 36(2):141–149, 2008.

[60] G. Dobson and S. Kalish. Positioning and pricing a product line. *Marketing Science*, 7(2):107–125, 1988.

[61] G. Dobson and S. Kalish. Heuristics for pricing and positioning a product-line using conjoint and cost data. *Management Science*, 39(2):160–175, 1993.

[62] L. F. Domínguez and E. N. Pistikopoulos. Multiparametric programming based algorithms for pure integer and mixed-integer bilevel programming problems. *Computers & Chemical Engineering*, 34(12):2097–2106, 2010.

[63] J.-P. Dussault, P. Marcotte, S. Roch, and G. Savard. A smoothing heuristic for a bilevel pricing problem. *European Journal of Operational Research*, 174(3):1396–1413, 2006.

[64] T. A. Edmunds and J. F. Bard. An algorithm for the mixed-integer nonlinear bilevel programming problem. *Annals of Operations Research*, 34(1):149–162, 1992.

[65] N. P. Faísca, V. Dua, B. Rustem, P. M. Saraiva, and E. N. Pistikopoulos. Parametric global optimisation for bilevel programming. *Journal of Global Optimization*, 38(4):609–623, 2007.

[66] J. E. Falk. A linear max—min problem. *Mathematical Programming*, 5(1):169–188, 1973.

[67] C. G. Fernandes, C. E. Ferreira, A. J. Franco, and R. C. Schouery. The envy-free pricing problem, unit-demand markets and connections with the network pricing problem. *Discrete Optimization*, 22:141–161, 2016.

[68] M. Fischetti, I. Ljubić, M. Monaci, and M. Sinnl. Intersection cuts for bilevel optimization. In *International Conference on Integer Programming and Combinatorial Optimization*, pages 77–88. Springer, 2016.

[69] M. Fischetti, I. Ljubic, M. Monaci, and M. Sinnl. Instances and solver software for mixed-integer bilevel linear problems, 2017.

[70] M. Fischetti, I. Ljubić, M. Monaci, and M. Sinnl. A new general-purpose algorithm for mixed-integer bilevel linear programs. *Operations Research*, 65(6):1615–1637, 2017.

[71] M. Fischetti, I. Ljubić, M. Monaci, and M. Sinnl. On the use of intersection cuts for bilevel optimization. *Mathematical Programming*, 172(1-2):77–103, 2018.

[72] P. Fontaine and S. Minner. Benders decomposition for discrete–continuous linear bilevel problems with application to traffic network design. *Transportation Research Part B: Methodological*, 70:163–172, 2014.

[73] J. Fortuny-Amat and B. McCarl. A representation and economic interpretation of a two-level programming problem. *Journal of the operational Research Society*, 32(9):783–792, 1981.

[74] B. Fu, C. Ouyang, C. Li, J. Wang, and E. Gul. An improved mixed integer linear programming approach based on symmetry diminishing for unit commitment of hybrid power system. *Energies*, 12(5):833, 2019.

[75] B. Goldengorin, D. Ghosh, and G. Sierksma. Branch and peg algorithms for the simple plant location problem. *Computers & Operations Research*, 31(2):241–255, 2004.

[76] I. E. Grossmann and C. A. Floudas. Active constraint strategy for flexibility analysis in chemical processes. *Computers & Chemical Engineering*, 11(6):675–693, 1987.

[77] Z. H. Gümüş and C. A. Floudas. Global optimization of mixed-integer bilevel programming problems. *Computational Management Science*, 2(3):181–212, 2005.

[78] A. Gupte, S. Ahmed, M. S. Cheon, and S. Dey. Solving mixed integer bilinear problems using milp formulations. *SIAM Journal on Optimization*, 23(2):721–744, 2013.

[79] V. Guruswami, J. D. Hartline, A. R. Karlin, D. Kempe, C. Kenyon, and F. McSherry. On profit-maximizing envy-free pricing. In *SODA*, volume 5, pages 1164–1173. Citeseer, 2005.

[80] W. J. Gutjahr and N. Dzubur. Bi-objective bilevel optimization of distribution center locations considering user equilibria. *Transportation Research Part E: Logistics and Transportation Review*, 85:1–22, 2016.

[81] S. D. Handoko, L. H. Chuin, A. Gupta, O. Y. Soon, H. C. Kim, and T. P. Siew. Solving multi-vehicle profitable tour problem via knowledge adoption in evolutionary bi-level programming. In *2015 IEEE Congress on Evolutionary Computation (CEC)*, pages 2713–2720. IEEE, 2015.

[82] P. Hanjoul and D. Peeters. A facility location problem with clients' preference orderings. *Regional Science and Urban Economics*, 17(3):451–473, 1987.

[83] P. Hansen, Y. Kochetov, and N. Mladenovi. *Lower bounds for the uncapacitated facility location problem with user preferences*. Groupe d'études et de recherche en analyse des décisions, HEC Montréal, 2004.

[84] L. Hecheng and W. Yuping. Exponential distribution-based genetic algorithm for solving mixed-integer bilevel programming problems. *Journal of Systems Engineering and Electronics*, 19(6):1157–1164, 2008.

[85] G. Heilporn, M. Labbé, P. Marcotte, and G. Savard. A polyhedral study of the network pricing problem with connected toll arcs. *Networks: An International Journal*, 55(3):234–246, 2010.

[86] S. R. Hejazi, A. Memariani, G. Jahanshahloo, and M. M. Sepehri. Linear bilevel programming solution by genetic algorithm. *Computers & Operations Research*, 29(13):1913–1925, 2002.

[87] A. J. Hoffman and J. B. Kruskal. Integral boundary points of convex polyhedra. In *50 Years of integer programming 1958-2008*, pages 49–76. Springer, 2010.

[88] X. Hu and D. Ralph. Convergence of a penalty method for mathematical programming with complementarity constraints. *Journal of Optimization Theory and Applications*, 123(2):365–390, 2004.

[89] IBM. Module callbacks, https://www.ibm.com/docs/en/icos/12.10.0?topic=manual-cplexcallbacks, 2010-09-30.

[90] E. Israeli and R. K. Wood. Shortest-path network interdiction. *Networks: An International Journal*, 40(2):97–111, 2002.

[91] R. G. Jeroslow. The polynomial hierarchy and a simple model for competitive analysis. *Mathematical programming*, 32(2):146–164, 1985.

[92] H. Jia, F. Ordóñez, and M. Dessouky. A modeling framework for facility location of medical services for large-scale emergencies. *IIE transactions*, 39(1):41–55, 2007.

[93] S. Kalish and P. Nelson. A comparison of ranking, rating and reservation price measurement in conjoint analysis. *Marketing Letters*, 2(4):327–335, 1991.

[94] P.-M. Kleniati and C. S. Adjiman. Branch-and-sandwich: a deterministic global optimization algorithm for optimistic bilevel programming problems. part i: Theoretical development. *Journal of Global Optimization*, 60(3):425–458, 2014.

[95] C. D. Kolstad. A review of the literature on bi-level mathematical programming. Technical report, Los Alamos National Laboratory Los Alamos, NM, 1985.

[96] U. G. Kraus and C. A. Yano. Product line selection and pricing under a share-of-surplus choice model. *European Journal of Operational Research*, 150(3):653–671, 2003.

[97] F. M. Kue. Mixed integer bilevel programming problems. 2017.

[98] M. Labbé, P. Marcotte, and G. Savard. A bilevel model of taxation and its application to optimal highway pricing. *Management science*, 44(12-part-1):1608–1622, 1998.

[99] M. Labbé, P. Marcotte, and G. Savard. On a class of bilevel programs. In *Nonlinear optimization and related topics*, pages 183–206. Springer, 2000.

[100] M. Labbé and A. Violin. Bilevel programming and price setting problems. *Annals of Operations Research*, 240(1):141–169, 2016.

[101] F. Legillon, A. Liefooghe, and E.-G. Talbi. Cobra: A cooperative coevolutionary algorithm for bi-level optimization. In *2012 IEEE Congress on evolutionary computation*, pages 1–8. IEEE, 2012.

[102] H. Li and Y. Wang. An evolutionary algorithm based on a new decomposition scheme for nonlinear bilevel programming problems. *IJCNS*, 3(1):87–93, 2010.

[103] Y. Liu, H. Li, and H. Chen. A genetic algorithm for solving linear integer bilevel programming problems. In *2018 14th International Conference on Computational Intelligence and Security (CIS)*, pages 40–44. IEEE, 2018.

[104] A. Lodi, T. K. Ralphs, and G. J. Woeginger. Bilevel programming and the separation problem. *Mathematical Programming*, 146(1-2):437–458, 2014.

[105] L. Lozano and J. Smith. A value-function-based exact approach for the bilevel mixed-integer programming problem. *Operations Research*, 65(3):768–786, 2017.

[106] S. Lucidi and F. Rinaldi. Exact penalty functions for nonlinear integer programming problems. *Journal of optimization theory and applications*, 145(3):479–488, 2010.

[107] Y. Lv, T. Hu, G. Wang, and Z. Wan. A penalty function method based on kuhn–tucker condition for solving linear bilevel programming. *Applied Mathematics and Computation*, 188(1):808–813, 2007.

[108] R. Mathieu, L. Pittard, and G. Anandalingam. Genetic algorithm based approach to bi-level linear programming. *RAIRO-Operations Research-Recherche Opérationnelle*, 28(1):1–21, 1994.

[109] G. P. McCormick. Computability of global solutions to factorable nonconvex programs: Part i—convex underestimating problems. *Mathematical programming*, 10(1):147–175, 1976.

[110] A. G. Mersha and S. Dempe. Linear bilevel programming with upper level constraints depending on the lower level solution. *Applied mathematics and computation*, 180(1):247–254, 2006.

[111] N. Mladenović, J. Brimberg, and P. Hansen. A note on duality gap in the simple plant location problem. *European Journal of Operational Research*, 174(1):11–22, 2006.

[112] J. Moore and J. Bard. An algorithm for the zero-one bilevel programming problem. *Department of Mechanical Engineering, University of Texas, Austin*, 1987.

[113] J. T. Moore and J. F. Bard. The mixed integer linear bilevel programming problem. *Operations research*, 38(5):911–921, 1990.

[114] I. Nishizaki, M. Sakawa, K. Niwa, and Y. Kitaguchi. A computational method using genetic algorithms for obtaining stackelberg solutions to two-level linear programming problems. *Electronics and Communications in Japan (Part III: Fundamental Electronic Science)*, 85(6):55–62, 2002.

[115] R. Oberdieck and E. N. Pistikopoulos. Explicit hybrid model-predictive control: The exact solution. *Automatica*, 58:152–159, 2015.

[116] R. Oberdieck, M. Wittmann-Hohlbein, and E. N. Pistikopoulos. A branch and bound method for the solution of multiparametric mixed integer linear programming problems. *Journal of Global Optimization*, 59(2-3):527–543, 2014.

[117] V. Oduguwa and R. Roy. Bi-level optimisation using genetic algorithm. In *Proceedings 2002 IEEE International Conference on Artificial Intelligence Systems (ICAIS 2002)*, pages 322–327. IEEE, 2002.

[118] U. of Southampton. The iridis compute cluster, 2021.

[119] A. Ouattara and A. Aswani. Duality approach to bilevel programs with a convex lower level. In *2018 Annual American Control Conference (ACC)*, pages 1388–1395. IEEE, 2018.

[120] J. V. Outrata. A note on the usage of nondifferentiable exact penalties in some special optimization problems. *Kybernetika*, 24(4):251–258, 1988.

[121] J. H. Owen and S. Mehrotra. On the value of binary expansions for general mixed-integer linear programs. *Operations Research*, 50(5):810–819, 2002.

[122] G. P. Papavassilopoulos. Algorithms for static stackelberg games with linear costs and polyhedra constraints. In *1982 21st IEEE Conference on Decision and Control*, pages 647–652. IEEE, 1982.

[123] S. P. Parvasi, M. Mahmoodjanloo, and M. Setak. A bi-level school bus routing problem with bus stops selection and possibility of demand outsourcing. *Applied Soft Computing*, 61:222–238, 2017.

[124] S. Pineda, H. Bylling, and J. Morales. Efficiently solving linear bilevel programming problems using off-the-shelf optimization software. *Optimization and Engineering*, 19(1):187–211, 2018.

[125] F. Plastria. Static competitive facility location: an overview of optimisation approaches. *European Journal of Operational Research*, 129(3):461–470, 2001.

[126] P.-L. Poirion, S. Toubaline, C. D'Ambrosio, and L. Liberti. Algorithms and applications for a class of bilevel milps. *Discrete Applied Mathematics*, 272:75–89, 2020.

[127] M. Qi, M. Xia, Y. Zhang, and L. Miao. Competitive facility location problem with foresight considering service distance limitations. *Computers & Industrial Engineering*, 112:483–491, 2017.

[128] A. Rahmani and S. MirHassani. Lagrangean relaxation-based algorithm for bilevel problems. *Optimization Methods and Software*, 30(1):1–14, 2015.

[129] A. Rahmani and M. Yousefikhoshbakht. An effective branch-and-cut algorithm in order to solve the mixed integer bi-level programming. *International Journal of Production Management and Engineering*, 5(1):1–10, 2017.

[130] D. Ralph* and S. J. Wright. Some properties of regularization and penalization schemes for mpecs. *Optimization Methods and Software*, 19(5):527–556, 2004.

[131] C. ReVelle, D. Bigman, D. Schilling, J. Cohon, and R. Church. Facility location: a review of context-free and ems models. *Health Services Research*, 12(2):129, 1977.

[132] F. Rinaldi. New results on the equivalence between zero-one programming and continuous concave programming. *Optimization Letters*, 3(3):377–386, 2009.

[133] J.-S. Roy. "binarize and project" to generate cuts for general mixed-integer programs. *Algorithmic Operations Research*, 2(1):37–51, 2007.

[134] G. K. Saharidis and M. G. Ierapetritou. Resolution method for mixed integer bi-level linear problems based on decomposition technique. *Journal of Global Optimization*, 44(1):29–51, 2009.

[135] S. Saranwong and C. Likasiri. Product distribution via a bi-level programming approach: Algorithms and a case study in municipal waste system. *Expert Systems with Applications*, 44:78–91, 2016.

[136] S. Scholtes. Convergence properties of a regularization scheme for mathematical programs with complementarity constraints. *SIAM Journal on Optimization*, 11(4):918–936, 2001.

[137] B. V. Sheela and P.Ramamoorthy. Swift—a new constrained optimization technique. *Computer Methods in Applied Mechanics and Engineering*, 6(3):309–317, 1975.

[138] H. D. Sherali and W. P. Adams. *A reformulation-linearization technique for solving discrete and continuous nonconvex problems*, volume 31. Springer Science & Business Media, 2013.

[139] R. Shioda, L. Tunçel, and T. G. Myklebust. Maximum utility product pricing models and algorithms based on reservation price. *Computational Optimization and Applications*, 48(2):157–198, 2011.

[140] S. Siddiqui and S. A. Gabriel. An sos1-based approach for solving mpecs with a natural gas market application. *Networks and Spatial Economics*, 13(2):205–227, 2013.

[141] A. Sinha, Z. Lu, K. Deb, and P. Malo. Bilevel optimization based on iterative approximation of multiple mappings. *Journal of Heuristics*, 26(2):151–185, 2020.

[142] A. Sinha, P. Malo, and K. Deb. Efficient evolutionary algorithm for single-objective bilevel optimization. *arXiv preprint arXiv:1303.3901*, 2013.

[143] A. Sinha, P. Malo, and K. Deb. An improved bilevel evolutionary algorithm based on quadratic approximations. In *2014 IEEE Congress on Evolutionary Computation (CEC)*, pages 1870–1877. IEEE, 2014.

[144] A. Sinha, P. Malo, and K. Deb. Evolutionary algorithm for bilevel optimization using approximations of the lower level optimal solution mapping. *European Journal of Operational Research*, 257(2):395–411, 2017.

[145] A. Sinha, P. Malo, and K. Deb. A review on bilevel optimization: from classical to evolutionary approaches and applications. *IEEE Transactions on Evolutionary Computation*, 22(2):276–295, 2017.

[146] A. Sinha, T. Soun, and K. Deb. Using karush-kuhn-tucker proximity measure for solving bilevel optimization problems. *Swarm and evolutionary computation*, 44:496–510, 2019.

[147] M. Sinnl. Bilevel integer programming and interdiction problems.

[148] J. Smith and Y. Song. A survey of network interdiction models and algorithms. *European Journal of Operational Research*, 283(3):797–811, 2020.

[149] K.-M. Steinborn-Busse. Mixed-integer non-linear bilevel pricing problem. https://gitlab.com/KSB_1871/minlbpp, 2022.

[150] D. Sun and L. Qi. On ncp-functions. *Computational Optimization and Applications*, 13(1):201–220, 1999.

[151] S. Tahernejad, T. K. Ralphs, and S. T. DeNegre. A branch-and-cut algorithm for mixed integer bilevel linear optimization problems and its implementation. *Mathematical Programming Computation*, 12(4):529–568, 2020.

[152] K. T. Talluri and G. J. Van Ryzin. The theory and practice of revenue management, vol. 68 springer science & business media. 2006.

[153] E. Towle. *Formulations and Valid Inequalities for Network Interdiction Problems and Reverse Convex Sets*. The University of Wisconsin-Madison, 2019.

[154] A. Tsoukalas, W. Wiesemann, B. Rustem, et al. Global optimisation of pessimistic bi-level problems. *Lectures on global optimization*, 55:215–243, 2009.

[155] H. Tuy, A. Migdalas, and P. Värbrand. A global optimization approach for the linear two-level program. *Journal of Global Optimization*, 3(1):1–23, 1993.

[156] L. Vicente, G. Savard, and J. Judice. Discrete linear bilevel programming problem. *Journal of optimization theory and applications*, 89(3):597–614, 1996.

[157] L. N. Vicente and P. H. Calamai. Bilevel and multilevel programming: A bibliography review. *Journal of Global optimization*, 5(3):291–306, 1994.

[158] H. F. von Stackelberg. *Market structure and equilibrium*. Springer Science & Business Media, 2010. Translated from the original *Marktform und Gleichgewicht*, Vienna 1934.

[159] L. Wang and P. Xu. The watermelon algorithm for the bilevel integer linear programming problem. *SIAM Journal on Optimization*, 27(3):1403–1430, 2017.

[160] Y. Wang, Y.-C. Jiao, and H. Li. An evolutionary algorithm for solving nonlinear bilevel programming based on a new constraint-handling scheme. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 35(2):221–232, 2005.

[161] Y. Wang, H. Li, and C. Dang. A new evolutionary algorithm for a class of nonlinear bilevel programming problems and its global convergence. *INFORMS Journal on Computing*, 23(4):618–629, 2011.

[162] A. Weber. *Theory of the Location of Industries*. University of Chicago Press, 1929.

[163] U.-P. Wen and S.-T. Hsu. Linear bi-level programming problems—a review. *Journal of the Operational Research Society*, 42(2):125–133, 1991.

[164] U.-P. Wen and A. Huang. A simple tabu search method to solve the mixed-integer linear bilevel programming problem. *European Journal of Operational Research*, 88(3):563–571, 1996.

[165] U.-P. Wen and Y. Yang. Algorithms for solving the mixed integer two-level linear programming problem. *Computers & Operations Research*, 17(2):133–142, 1990.

[166] D. J. White and G. Anandalingam. A penalty function approach for solving bi-level linear programs. *Journal of Global Optimization*, 3(4):397–419, 1993.

[167] W. Wiesemann, A. Tsoukalas, P.-M. Kleniati, and B. Rustem. Pessimistic bilevel optimization. *SIAM Journal on Optimization*, 23(1):353–380, 2013.

[168] R. Wollmer. Removing arcs from a network. *Operations Research*, 12(6):934–940, 1964.

[169] K. Wood. Bilevel network interdiction models: Formulations and solutions. *Wiley encyclopedia of operations research and management science*, 2010.

[170] Y. Wu, T. Xu, H. Meng, W. Wei, S. Cai, and L. Guo. Energy storage capacity allocation for distribution grid applications considering the influence of ambient temperature. *IET Energy Systems Integration*, 4(1):143–156, 2022.

[171] J. Xie, Y. Mei, A. T. Ernst, X. Li, and A. Song. A bi-level optimization model for grouping constrained storage location assignment problems. *IEEE transactions on cybernetics*, 48(1):385–398, 2016.

[172] P. Xu. Three essays on bilevel optimization algorithms and applications. 2012.

[173] J. J. Ye and D. Zhu. Optimality conditions for bilevel programming problems. *Optimization*, 33(1):9–27, 1995.

[174] Y. Yin. Genetic-algorithms-based approach for bilevel programming models. *Journal of transportation engineering*, 126(2):115–120, 2000.

[175] D. Yue, J. Gao, B. Zeng, and F. You. A projection-based reformulation and decomposition algorithm for global optimization of a class of mixed integer bilevel linear programs. *Journal of Global Optimization*, 73(1):27–57, 2019.

[176] H. Zare, J. S. Borrero, B. Zeng, and O. A. Prokopyev. A note on linearized reformulations for a class of bilevel linear integer problems. *Annals of Operations Research*, 272(1-2):99–117, 2019.

[177] A. B. Zemkoho and S. Zhou. Theoretical and numerical comparison of the karush–kuhn–tucker and value function reformulations in bilevel optimization. *Computational Optimization and Applications*, 78(2):625–674, 2021.

[178] B. Zeng and Y. An. Solving bilevel mixed integer program by reformulations and decomposition. *Optimization online*, pages 1–34, 2014.

[179] R. Zenklusen. Matching interdiction. *Discrete Applied Mathematics*, 158(15):1676–1690, 2010.

[180] R. Zhang, T. Jiang, G. Li, X. Li, and H. Chen. Stochastic optimal energy management and pricing for load serving entity with aggregated tcls of smart buildings: A stackelberg game approach. *IEEE Transactions on industrial informatics*, 17(3):1821–1830, 2020.

[181] Y. Zheng, G. Zhang, Z. Zhang, and J. Lu. A reducibility method for the weak linear bilevel programming problems and a case study in principal-agent. *Information Sciences*, 454:46–58, 2018.