2012

Three essays on bilevel optimization algorithms and applications

Pan Xu
Iowa State University

Three essays on bilevel optimization algorithms and applications

by

Pan Xu

A dissertation submitted to the graduate faculty

in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Major: Industrial Engineering

Program of Study Committee:

Lizhi Wang, Major Professor

William D. Beavis

Guiping Hu

Sarah Ryan

Giora Slutzki

Srikanta Tirthapura

Iowa State University

Ames, Iowa

2012

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACKNOWLEDGEMENTS

It is a pleasure to thank the many people who made this thesis possible. It is difficult to overstate my gratitude to my Ph.D. supervisor, Dr. Lizhi Wang. With his enthusiasm, his inspiration, and his great efforts to explain things clearly and simply, he helped to make research fun for me. Throughout my thesis-writing period, he provided encouragement, sound advice, good teaching, and lots of good ideas. I would have been lost without him.

This thesis could not have been completed without the tremendous help from other committee members: Dr. Beavis, Dr. Hu, Dr. Ryan, Dr. Slutzki and Dr. Tirthapura. I feel much indebted for their time and effort reading the thesis, and providing me comments that greatly improved it. Needless to say, I am the only person who is responsible for any error contained in this thesis.

The financial support from the Miller Fellowship is also gratefully acknowledged.

I am indebted to many student colleagues for providing a stimulating and fun environment in which I learn and grow. I am especially grateful to Zhaoyang Duan, Yanyi He, Nan Gao, Shan Jin, Xiaoli Yang, Youngrok Lee, Chenlu Lou, Wenbo Shi, Rui Yang, Peihan Zhong and Kristopher Watts at ISU. Zhaoyang, Yanyi and Shan are particularly helpful in teaching me how to lead a wonderful life in Ames.

I also feel thankful to Bill Rowcliffe and Austin Sullivan who gave me many tips regarding how to drive a car, how to fish in America and how to adapt to the multi-valued American culture.

ABSTRACT

This thesis consists of three journal papers I have worked on during the past three years of my PhD studies.

In the first paper, we presented a multi-objective integer programming model for the gene stacking problem. Although the gene stacking problem is proved to be NP-hard, we have been able to obtain Pareto frontiers for smaller sized instances within one minute using the state-of-the-art commercial computer solvers in our computational experiments.

In the second paper, we presented an exact algorithm for the bilevel mixed integer linear programming (BMILP) problem under three simplifying assumptions. Compared to these existing ones, our new algorithm relies on weaker assumptions, explicitly considers infinite optimal, infeasible, and unbounded cases, and is proved to terminate infinitely with the correct output. We report results of our computational experiments on a small library of BMILP test instances, which we have created and made publicly available online.

In the third paper, we presented the watermelon algorithm for the bilevel integer linear programming (BILP) problem. To our best knowledge, it is the first exact algorithm which promises to solve all possible BILPs, including finitely optimal, infeasible, and unbounded cases. What is more, our algorithm does not rely on any simplifying condition, allowing even the case of unboundedness for the high point problem. We prove that the watermelon algorithm must finitely terminate with the correct output. Computational experiments are also reported showing the efficiency of our algorithm.

CHAPTER 1. GENERAL INTRODUCTION

1.1 Introduction

This thesis consists of three journal papers I have worked on during the past three years of my PhD studies. The research presented in these papers centers on the following two topics: (1) the application of a multi-objective optimization model in the gene stacking problem; (2) algorithm design for solving the bilevel integer/mixed integer linear programming problem. This chapter briefly introduces the background of the dissertation topics. Chapters 2 to 4 are the three papers, respectively. Chapter 5 summarizes the contributions of my dissertation and discusses some future research directions.

1.1.1 Multi-Objective Optimization

Multi-Objective optimization naturally arises in the practical cases when we consider several objectives simultaneously while making a decision. Generally, when all of the constraints and objectives involved can be expressed in terms of linear inequalities and linear functions, a multi-objective optimization model is formulated as a multi-objective linear programming (MOLP) problem as follows:

$$\max_{x} \quad \zeta = [c_1^\top x, c_2^\top x, \cdots, c_m^\top x]^\top \tag{1.1}$$

$$\text{s.t.} \quad Ax \leq b, \tag{1.2}$$

$$x \geq 0. \tag{1.3}$$

In a multi-objective optimization problem, optimality of all objectives is usually not achievable at the same time. An important concept for making wise tradeoffs among competing

objectives is *Pareto optimality*. A solution is Pareto optimal when it is not dominated by any feasible solution. Mathematically, Pareto optimality for MOLP (1.1)-(1.3) can be defined as follows:

Definition 1. *Let* $X = \{x | x \geq 0, Ax \leq b\}$. *A solution* x *is Pareto optimal in* (1.1)-(1.3) *if and only if: (1)* $x \in X$*; (2) there is no solution* $y \in X$ *such that* $c_i^\top y \geq c_i^\top x, \forall 1 \leq i \leq m$ *and* $\exists i \in \{1, 2, \cdots, m\}, c_i^\top y > c_i^\top x$.

According to the definition above, there could exist multiple Pareto optimal solutions, and the collection of these solutions, called Pareto frontier, yields a range of good tradeoff options for specific decision makers to choose from. Any of these solutions can be claimed optimal, depending on specific situations or perspectives.

In practical applications, we encounter more situations when variables take integral value only rather than continuous. In that case, multi-objective discrete optimization is formulated by a multi-objective integer programming (MOIP) with (1.3) replaced by $x \geq 0, x \in \mathbb{Z}^n$. The concepts such as Pareto optimality and Pareto frontier can be naturally inherited from those in the continuous case. As for a multi-objective discrete optimization, in most cases, the size of Pareto frontier can be exponentially large or infinite and consequently it is almost computationally intractable to compute the whole Pareto frontier. Note that even for the simplest MOLP with two objectives, it is $\mathcal{NP}$-hard to identity if a feasible solution belong to the Pareto frontier (Papadimitriou and Yannakakis, 2000).

Multi-objective optimization models are widely used to capture the trade-offs between two or more conflicting objectives under a certain set of constraints. Its applications can be found in structural design (Tseng and Lu, 1990), chemical engineering (Rajesh et al., 2001), computer and software engineering (Blickle et al., 1998), finance (Bouri et al., 2002), etc.

1.1.2 Gene Stacking Problem

One hundred and fifty years ago Charles Darwin described how adaptation by a few founders to a new environment could result in emergence of a new species. In his revolutionary book, The Origin of Species (Darwin, 1872), he described how this idea followed logically from the

experience of animal breeding. Fifty years ago, Alan Robertson described the impact of intense selection on the progeny from a small founder population: An unadaptive population of sub-optimal phenotypes. To ameliorate such undesirable consequences, new sources of desirable genetic variability need to be introduced without disrupting genetic networks assembled by generations of selection. Currently this is attempted through ad hoc procedures with occasional success. In this paper, we present an optimization model for stacking desirable sources of genetic variability into adapted breeding populations.

Plant breeders have shown that introgression of single genes into existing inbred lines through marker assisted backcrossing is trivial (Anderson et al., 2008; Cahill and Schmidt, 2004; Concibido et al., 1994; Eathington et al., 2007; Pumphrey et al., 2007; Tanksley and Hewitt, 1988) and the ability to pyramid desirable alleles for an oligogenic trait into a single background is straight-forward (Bonnet et al., 2005; Eathington et al., 2007; Howes et al., 1998; Wang et al., 2007b). However, for complex traits, it is not obvious if there is a single best marker assisted breeding method. Simulation modeling is a tool that can be employed to address this question (Podlich et al., 2004), and results indicate that optimal breeding strategies are dependent upon the genetic architecture of the trait, reproductive biology of the crop and the size and structure of the breeding population. As challenging as marker assisted breeding is for a single complex trait, the challenge is greater for multiple breeding objectives involving several complex traits. A heuristic algorithm is presented in Servin et al. (2004) to combine a series of target genes identified in different parents into a single genotype, on the basis of both the population sizes they require and on their total duration. Other constraints could also be taken into account, but it would further complicate the issue and require more advanced methodologies. For example, consider the objective of maintaining genetic diversity throughout the genome while applying selection on alleles at a large number of loci scattered throughout the genome.

In the last decades operations research (Hillier and Lieberman, 2005) has been addressing these types of challenges in industrial engineering, manufacturing systems, financial systems and business management using deterministic (Murty, 1995), stochastic (Birge and Louveaux, 1997; Kall and Wallace, 1994), and dynamic programming (Bellman, 1957; Puterman, 1994)

methods. Given the emergence of information on plant genomes, we hypothesize that operations research tools can now be used for large scale genetic improvement of crops.

In Chapter 2, we attempt to address the problem of gene stacking using a multi-objective integer programming approach. From a population genetic perspective, the goal represents selection and fixation of desirable alleles at a set of loci responsible for the desirable trait, while maintaining genetic variability at all remaining loci in an adapted breeding population. From the perspective of operations research, the challenge is to determine the Pareto frontier of an NP-hard optimization problem with two competing objectives. Merging these two perspectives produces the following goal: Determine the Pareto frontier of marker assisted breeding strategies with respect to the likelihood of success and the number of generations while maintaining the genetic diversity of the adapted population.

1.1.3 Bilevel Optimization

In traditional linear programming programs, there is only one single decision-making party who tries to figure out an optimal decision subject to a certain set of constraints such that their objectives reach the best. As opposed to that, in a bilevel program, there are typically two competing decision-making parties: one is upper level decision makers and the other is lower level decision makers. The two levels interact with each other in the following ways: (1) the set of lower level constraints is completely determined by the upper level's decision; (2) upper level objectives are mutually determined by decisions from both levels; (3) for each decision made by the upper level, lower level will choose the best option according to their objectives and the updated set of constraints. From the first and third point, we can see that decision from the upper level inexplictly determine lower level's best reaction. Thus we can view a bilevel program as a one-level traditional optimization program in which: (1) upper level is the only decision-making party and (2) upper level try to control the lower level's decision in the way that lower level will react by choosing the best option under the corresponding constraints. The problem captured in a bilevel program is how does the upper level make a decision such that their objectives can reach the best.

Bilevel optimization models have vast applications in the real world. Brotcorne et al. (2001);

Cote et al. (2003); Labbe et al. (1998) presented applications of bilevel optimization models into transportation area where upper level represents decision makers setting price or tolls while lower level represents users traveling on shortest path within budget. Salmeron et al. (2004) showed an application into national security where upper level represents defenders allocating resources to minimize risk while lower level represents offenders trying maximizing destruction for a given budget. Fampa et al. (2008) offered an application into energy where upper level is the energy provider minimizing total cost while lower level is the energy consumer determining the pattern of consumption. Bard et al. (2000) presented an application into taxation policy where upper level is the policy maker maximizing social welfare and lower level is the producer maximizing profit. There are applications into other areas as well such as network interdiction (Morton et al., 2007; Lim and Smith, 2007), national agriculture planning (Fortuny-Amat and McCarl, 1981), decentralized management of multidivisional firms (Bard, 1983), etc.

Mathematically, a general bilevel optimization model can be formulated as a bilevel mixed integer linear programming (BMILP) as follows:

$$\max_{x,y} \quad \zeta_1 = c^\top x + d_1^\top y \tag{1.4}$$

$$\text{s. t.} \quad A_1 x + B_1 y \leq b_1, \tag{1.5}$$

$$y \in \text{argmax}_{\tilde{y}}\{\zeta_2 = d_2^\top \tilde{y} : A_2 x + B_2 \tilde{y} \leq b_2, \tilde{y} \geq 0, \tilde{y}_J \in \mathbb{Z}^{|J|}\}, \tag{1.6}$$

$$x \geq 0, x_I \in \mathbb{Z}^{|I|}. \tag{1.7}$$

BMILP problems (1.4)-(1.7) are hard. The computational challenges mainly lie in the following points: (1) the bilevel feasible region can be noncovex; (2) the optimal value can be unattainable; (3) the unboundedness is hard to detect; (4) the optimal value obtained after relaxing variables in y to be continuous in (1.6) fails to serve as an upper bound. To demonstrate these challenges, let us consider the example offered in Köppe et al. (2009).

Example 1.

$$\begin{aligned} \max_{x,y} \quad & \zeta_1 = -x + y \\ \text{s. t.} \quad & 0 \leq x \leq 1, \\ & y \in \text{argmax}_{\tilde{y}}\{\zeta_2 = -\tilde{y} : 0 \leq \tilde{y} \leq 1; \tilde{y} \geq x; \tilde{y} \in \mathbb{Z}\}. \end{aligned}$$

Easily we can check the following facts: (1) the bilevel feasible region is $\{(x, y)|(0 < x \leq 1, y = 1) \vee (x = 0, y = 0)\}$ which is nonconvex; (2) the supreme of objective value is 1 which can not be attained; (3) the optimal value obtained after relaxing variables in y to be continuous is 0 which is not an upper bound.

Perhaps due to these challenges, there has been insufficient attention and progress on theoretical and algorithmic development. We were able to find only a handful of BMILP algorithms. In their seminal work (Moore and Bard, 1990), Moore and Bard presented perhaps the first branch-and-bound algorithms for BMILP under the assumption that $B_1 = 0$. The nature of the algorithms is heuristic, which does not guarantee their correctness or finite termination. However, they proved that the algorithms find the optimal solution if either all upper level variables x are integral or all lower level variables y are continuous. The case of unbounded BMILP is not discussed. DeNegre and Ralphs (2009) invented another algorithm under the assumptions that variables in both levels are all integral, and $B_1 = 0$. No proof of correctness and finite termination is provided, nor is the unbounded case discussed. Relevant literature also includes Köppe et al. (2009) and Dempe (2001). Köppe et al. (2009) is among the first to discover that, if the upper level contains continuous variables and the lower level discrete ones, then the supremum of the BMILP may not be attainable even if it finitely exists. However, the paper only proved the existence of an algorithm that solves BMILP in polynomial time when n_2 is fixed, referring to parametric integer programming approaches (Eisenbrand and Shmonin, 2008). The algorithm proposed by Dempe (2001), although referred to as a discrete bilevel programming algorithm, solves a different model, which is more similar to what is referred to as the inverse mixed integer linear programming problem in Wang (2009).

In Chapter 3, we present an exact algorithm for the bilevel mixed integer linear programming (BMILP) problem under three simplifying assumptions: (1) variables in the upper level are all integral; (2) the feasible region for the upper level variables is bounded; (3) A_2 in (1.6) is integral. The algorithm that we present here differs from existing ones in the following four aspects. First, we allow the B_1 matrix in (1.5) to be nonzero. This significantly raises the difficulty of the problem, since obtaining a bilevel feasible solution is no longer straightforward.

Nevertheless, allowing for nonzero B_1 could be practically imperative, because in many real world situations the consequences of the lower level's decisions must be taken into account by the upper level as explicit constraints. Second, we allow the lower level to have both continuous and discrete variables. Third, we explicitly consider all possible outcomes of a BMILP, be it infeasible, unbounded, or finitely optimal. Fourth, we prove that our algorithm will finitely terminate with the correct output.

In Chapter 4, we present the so-called watermelon algorithm for the bilevel integer linear programming (BILP) problem. In this case, we assume variables in both levels are all integral. The watermelon algorithm we present in this paper differs from previous algorithms by having the following four features: (1) solves all BILP instances without any additional simplifying assumptions, (2) terminates in a finite number of iterations, (3) correctly identifies unbounded or infeasible instances, and (4) obtains a global optimal solution if one exists. Note that any BILP instance has only three possible outcomes: infeasible, unbounded, or optimal (having a global optimal solution).

CHAPTER 2. AN OPTIMIZATION APPROACH TO GENE STACKING

Pan Xu, Lizhi Wang, and William Beavis, "An optimization approach to gene stacking", *European Journal of Operational Research*, vol. 214(1), p. 168-178, 2011.

2.1 Abstract

We present a multi-objective integer programming model for the gene stacking problem, which is to bring desirable alleles found in multiple initial inbred lines to a single genotype. Pareto optimal solutions from the model provide strategic breeding schemes to maximize the likelihood of successfully creating the target genotypes and to minimize the number of generations needed to do so. The biological diversity consideration is also incorporated in the models to preserve desirable alleles of all variations in the target population. Although the gene stacking problem is proved to be NP-hard, we have been able to obtain Pareto frontiers for smaller sized instances within one minute using the state-of-the-art commercial computer solvers in our computational experiments.

2.2 Introduction

One hundred and fifty years ago Charles Darwin described how adaptation by a few founders to a new environment could result in emergence of a new species. In his revolutionary book, The Origin of Species (Darwin, 1872), he described how this idea followed logically from the experience of animal breeding. Fifty years ago, Alan Robertson described the impact of intense selection on the progeny from a small founder population: an unadaptive population of sub-optimal phenotypes. To ameliorate such undesirable consequences, new sources of desirable genetic variability need to be introduced without disrupting genetic networks assembled by

generations of selection. Currently this is attempted through ad hoc procedures with occasional success. In this paper, we present an optimization model for stacking desirable sources of genetic variability into adapted breeding populations.

Plant breeders have shown that introgression of single genes into existing inbred lines through marker assisted backcrossing is trivial (Anderson et al., 2008; Cahill and Schmidt, 2004; Concibido et al., 1994; Eathington et al., 2007; Pumphrey et al., 2007; Tanksley and Hewitt, 1988) and the ability to pyramid desirable alleles for an oligogenic trait into a single background is straight-forward (Bonnet et al., 2005; Eathington et al., 2007; Howes et al., 1998; Wang et al., 2007b). However, for complex traits, it is not obvious if there is a single best marker assisted breeding method. Simulation modeling is a tool that can be employed to address this question (Podlich et al., 2004), and results indicate that optimal breeding strategies are dependent upon the genetic architecture of the trait, reproductive biology of the crop and the size and structure of the breeding population. As challenging as marker assisted breeding is for a single complex trait, the challenge is greater for multiple breeding objectives involving several complex traits. A heuristic algorithm is presented in Servin et al. (2004) to combine a series of target genes identified in different parents into a single genotype, on the basis of both the population sizes they require and on their total duration. Other constraints could also be taken into account, but it would further complicate the issue and require more advanced methodologies. For example, consider the objective of maintaining genetic diversity throughout the genome while applying selection on alleles at a large number of loci scattered throughout the genome.

In the last decades, operations research (Hillier and Lieberman, 2005) has been addressing these types of challenges in industrial engineering, manufacturing systems, financial systems and business management using deterministic (Murty, 1995), stochastic (Birge and Louveaux, 1997; Kall and Wallace, 1994), and dynamic programming (Bellman, 1957; Puterman, 1994) methods. Given the emergence of information on plant genomes, we hypothesize that operations research tools can now be used for large scale genetic improvement of crops. In this paper, we attempt to address the problem of gene stacking using a multi-objective integer pro-

gramming approach. From a population genetic perspective, the goal represents selection and fixation of desirable alleles at a set of loci responsible for the desirable trait, while maintaining genetic variability at all remaining loci in an adapted breeding population. From the perspective of operations research, the challenge is to determine the Pareto frontier of an NP-hard optimization problem with two competing objectives. Merging these two perspectives produces the following goal: Determine the Pareto frontier of marker assisted strategies with respect to likelihood of success and the number of generations while maintaining the genetic diversity of the adapted population.

The remainder of this paper is organized as follows. Section 4.2 presents a multi-objective integer program formulation for the gene stacking problem, Section 4.3 presents both exact and heuristic algorithms for solving the model, Section 4.4 demonstrates the approach using two numerical examples, and Section 2.6 concludes the paper.

2.3 Problem Formulation

2.3.1 Mathematical Description of Breeding Rules

Consider the genetic architecture consisting of a single allele at each of M independently segregating loci. For the specific purpose of gene stacking, we simplify the biological characteristics of all alleles by classifying them as either desirable (denoted by 1) or undesirable (denoted by 0). Given a breeding population consisting of N inbreds, our goal is to bring all desirable alleles into a single inbred by strategically selecting inbreds from the breeding population for crossing. If we represent the breeding population with an M by N binary matrix indicating the desirability of the genotypes at each locus, then the goal becomes to develop an inbred line consisting of 1's at all loci (rows) in a single individual (column). When two inbreds are crossed, the progeny inherits a genome either from one parent as a whole or as a recombination of both parents'. To develop the ideal inbred (or target genotype), recombinations are usually

inevitable. For example $\begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} \times \begin{bmatrix} \boldsymbol{1} \\ \boldsymbol{1} \\ \boldsymbol{0} \end{bmatrix} \Rightarrow \begin{bmatrix} \boldsymbol{1} \\ \boldsymbol{1} \\ 1 \end{bmatrix}$ is a possible recombination outcome, which inherits two desirable alleles at the top two loci from one parent and one desirable allele at the bottom locus from the other parent. In this example, different fonts are used to differentiate alleles of the two parents, but desirable alleles at the same locus are assumed to be biological homogeneous.

2.3.2 Stacking Strategy and Efficiency Measure

There may exist a wide range of stacking strategies that could bring all desirable alleles to a single individual, and we attempt to find the most efficient one. We measure the efficiency of a strategy by two criteria: the likelihood of successfully obtaining the target genotype and the number of generations it takes to achieve it. For example, consider the initial population with five inbreds $\begin{bmatrix} 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 \end{bmatrix}$, named (1), (2), (3), (4), and (5) from left to right. To create the ideal inbred with six 1's in a column, one strategy is to cross (1) with (2), which requires five recombination points and one generation. Alternatively, we could also cross (3) and (4) first, and then cross their progeny that inherits all the 1's with (5), which will require two recombination points and two generations. The first strategy dominates the second one in terms of the number of generations but has a lower likelihood of success (due to a larger number of necessary rare recombinants), thus a tradeoff between probability and time is inevitable in this case. We will use a multi-objective approach to address the tradeoff between these two competing objectives.

2.3.3 Multi-Objective Optimization Model

In this section, we introduce a multi-objective optimization model to obtain the most efficient stacking strategy for a given breeding population. Let a binary matrix $A \in \mathbb{B}^{M\times N}$ denote an initial breeding population of N inbreds, each consisting of M alleles of interest that are represented by M rows. The probability of recombination between $A_{i,j}$ and $A_{i+1,j}$ is assumed to be r_i, which is constant across all inbreds. We introduce three binary decision variables $x \in \mathbb{B}^{M\times N}$, $y \in \mathbb{B}^{(M-1)\times N}$, and $z \in \mathbb{B}^{N\times 1}$. The variable $x_{i,j}$ indicates whether ($x_{i,j} = 1$) or not ($x_{i,j} = 0$) the corresponding allele $A_{i,j}$ is used in the ideal inbred; $y_{i,j}$ indicates whether ($y_{i,j} = 1$) or not ($y_{i,j} = 0$) there is a recombination point between $x_{i,j}$ and $x_{i+1,j}$; and z_j indicates whether ($z_j = 1$) or not ($z_j = 0$) the initial inbred j should contribute one or more desirable alleles to the ideal inbred.

We formulate the gene stacking problem as the following multi-objective optimization model:

$$\max_{x,y,z} \quad \rho = \sum_{i,j} y_{i,j} \log\left(\frac{r_i}{1-r_i}\right) \tag{2.1}$$

$$\min_{x,y,z} \quad \tau = \left\lceil \log_2 \sum_j z_j \right\rceil \tag{2.2}$$

$$\text{s.t.} \quad x_{i,j} \le A_{i,j}, \forall i,j \tag{2.3}$$

$$\sum_j x_{i,j} = 1, \forall i \tag{2.4}$$

$$y_{i,j} \ge x_{i,j} - x_{i+1,j}, \forall i,j \tag{2.5}$$

$$M \cdot z_j \ge \sum_i x_{i,j}, \forall j \tag{2.6}$$

$$x_{i,j}, y_{i,j}, z_j \in \{0,1\}, \forall i,j. \tag{2.7}$$

The first objective (2.1) is to maximize the likelihood of successfully obtaining the ideal inbred under the stacking strategy outlined in (x, y, z). Under the assumption that all recombination points occur independently, we can measure the likelihood of success by multiplying the probabilities r_i of all recombination points identified in y and those $1-r_i$ of all other points

in the genotype:

$$\prod_{i=1}^{M-1} \left[r_i^{\sum_j y_{i,j}} \cdot (1-r_i)^{1-\sum_j y_{i,j}} \right]. \tag{2.8}$$

We use this likelihood as one objective function to maximize. Notice that the objective function (2.1) is not maximizing this likelihood directly but its logarithm $\sum_{i,j} \log\left(\frac{r_i}{1-r_i}\right) y_{i,j} + \sum_i \log(1-r_i)$ less the constant term $\sum_i \log(1-r_i)$ that does not affect the determination of the optimal stacking strategy. Since the logarithm function is strictly increasing, maximizing objective (2.1) is precisely equivalent to maximizing the original likelihood (2.8), but the former linear function is much more computationally convenient.

The second objective (2.2) is to minimize the number of generations it takes to obtain the ideal inbred under the stacking strategy outlined in (x, y, z). The term $\sum_j z_j$ calculates the number of inbreds from the initial breeding population that are needed to create the ideal inbred. Since all inbreds can be crossed in pairs in each generation, taking the logarithm of $\sum_j z_j$ to the base 2 gives the number of necessary generations. To round up the number of generations to an integer in case of fractions, we use the ceiling function in objective (2.2), which gives the smallest integer that is larger than or equal to the number inside the ceiling operator.

Constraint (2.3) means that only desirable alleles can be used in the ideal inbred; Constraint (2.4) means that each allele of the ideal inbred comes from only one locus of one inbred; Constraint (2.5) flags a recombination point whenever $x_{i,j} = 1$ and $x_{i+1,j} = 0$; Constraint (2.6) indicates whether or not initial inbred j is used to create the ideal one; and Constraint (2.7) requires that all decision variables be binary.

2.3.4 Extension to Genetic Diversity

The formulation in Section 2.3.3 is based on the assumption that all alleles of interest can be classified as either desirable or undesirable. However, there are situations in which the exact biological effects of different alleles at a locus are not understood, thus it would be appropriate and necessary to maintain all variations in the target population rather than to arbitrarily keep

one and eliminate all others. To accommodate this requirement, we modify the model in Section 2.3.1 as follows. We classify an allele using non-negative integer numbers $\{0, 1, 2, ...\}$. A 0 means that the allele is known to be undesirable, and positive integers indicate the alleles that are either desirable or unknown but need to be preserved. Alleles of the same variations are also assumed to be biologically homogeneous thus do not need to be maintained in redundancy. The goal of gene stacking is consequently changed from creating one ideal inbred to creating a target population of inbreds such that all variations of all alleles are maintained but no undesirable ones are admitted. For example, if the initial breeding population is $\begin{bmatrix} 1 & 0 & 2 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 2 & 0 & 1 \\ 0 & 3 & 1 & 2 \end{bmatrix}$, then a target population could be $\begin{bmatrix} 1 & 2 & 2 \\ 1 & 1 & 1 \\ 1 & 2 & 1 \\ 2 & 3 & 1 \end{bmatrix}$. With the diversity extension, the problem adds another level of complexity: we need to design a target population first, and then a strategy for creating every single inbred in the target population.

Here we extend the multi-objective optimization model (2.1)-(2.7) to address the biological diversity requirement. Let a non-negative integer matrix $A \in \mathbb{Z}_+^{M\times N}$ denote an initial breeding population of inbreds, each consisting of M alleles of interest that are represented by M rows. The probability of recombination between $A_{i,j}$ and $A_{i+1,j}$ is assumed to be r_i. We also assume that the number of inbreds in the target population is known to be at most K, a given parameter. We introduce four binary decision variables $x \in \mathbb{B}^{M\times N\times K}$, $y \in \mathbb{B}^{(M-1)\times N\times K}$, $z \in \mathbb{B}^{N\times K}$, and $w \in \mathbb{B}^{K\times 1}$. The variables $x_{i,j,k}$, $y_{i,j,k}$, and $z_{j,k}$ are similar to the respective $x_{i,j}$, $y_{i,j}$, and z_j defined for model (2.1)-(2.7); the additional subscript k indicates that they contribute to the kth inbred in the target population. Variable w_k indicates whether ($w_k = 1$) or not ($w_k = 0$) target inbred k is necessary. If the model determines one or more w_k should be 0, then the actual number of inbreds in the target population may be less than K. The multi-objective

optimization model becomes the following:

$$\max_{x,y,z,w} \quad \rho = \sum_{i,k} w_k \log(1 - r_i) + \sum_{i,j,k} y_{i,j,k} \log\left(\frac{r_i}{1 - r_i}\right) \tag{2.9}$$

$$\min_{x,y,z,w} \quad \tau = \max_k \left\lceil \log_2 \sum_j z_{j,k} \right\rceil \tag{2.10}$$

$$\text{s. t.} \quad x_{i,j,k} \leq A_{i,j}, \forall i, j, k \tag{2.11}$$

$$\sum_j x_{i,j,k} = w_k, \forall i, k \tag{2.12}$$

$$y_{i,j,k} \geq x_{i,j,k} - x_{i+1,j,k}, \forall i, j, k \tag{2.13}$$

$$M \cdot z_{j,k} \geq \sum_i x_{i,j,k}, \forall j, k \tag{2.14}$$

$$\sum_{j,k:A_{i,j,k}=v} x_{i,j,k} \geq 1, \forall i, v \tag{2.15}$$

$$x_{i,j,k}, y_{i,j,k}, z_{j,k}, w_k \in \{0, 1\}, \forall i, j, k. \tag{2.16}$$

The first objective (2.9) is similar to (2.1), except that the likelihood is estimated by multiplying the recombination probabilities of all target inbreds in the population:

$$\prod_{i=1}^{M-1} \left[r_i^{\sum_{j,k} y_{i,j,k}} \cdot (1 - r_i)^{\sum_k w_k - \sum_{j,k} y_{i,j,k}} \right]. \tag{2.17}$$

The objective (2.9) is precisely its logarithm function. Since each target inbred takes a number of generations to obtain, the second objective (2.10) minimizes the largest one. Constraints (2.11), (2.13), (2.14), and (2.16) are the multiple-inbred version of (2.3), (2.5), (2.6), and (2.7), respectively. Constraint (2.12) is similar to (2.4) except that it takes into account the fact that not all K target inbreds are necessary; if $w_k = 0$, then $x_{i,j,k} = 0, \forall i, j$. Constraint (2.15) requires that all variations of desirable alleles at all loci be preserved in the target population. It is straightforward to check that when there is only one variation at each locus, the model (2.9)-(2.16) for the biological diversity case reduces to (2.1)-(2.7) for the single variation case in Section 2.3.3.

2.3.5 Complexity of the Problem

The multi-objective optimization model (2.9)-(2.16) is a hard problem. In the following, we show that even if we ignore the second objective (2.10), the resulting single objective problem

is NP-hard.

Proposition 1. *The gene stacking problem defined by (2.9) and (2.11)-(2.16) is NP-hard.*

Proof. We reduce the NP-hard set covering problem to the gene stacking problem. The set covering problem is equivalent to the integer program $\min_t \left\{\sum_i t_i : Ht \geq 1, t \in \mathbb{B}^{n\times 1}\right\}$, where $H \in \mathbb{B}^{m\times n}$ is a binary matrix. For a given H, we construct an initial breeding population matrix $A = \begin{bmatrix} 2-H & 2_{m\times 1} \\ 0_{1\times n} & 1 \end{bmatrix} \in \{0,1,2\}^{(m+1)\times(n+1)}$. Let $K = n+1$ and the recombination probability $r_m = 0.4$. For $1 \leq i \leq m-1$, we set r_i to be extremely small positive values such that recombination at any of these loci is always prohibitive. The resulting gene stacking problem is then equivalent to the set covering problem in the sense that a solution t is optimal to the set covering problem if and only if the following solution is optimal to the gene stacking problem: $x_{i,j,k} = \begin{cases} t_k, & \text{if } (i \leq m, j = k) \text{ or } (i = m+1, j = n+1); \\ 0 & \text{otherwise.} \end{cases}, \forall k = 1, ..., n,$ and $x_{i,j,n+1} = \begin{cases} 1, & \text{if } j = n+1; \\ 0, & \text{otherwise.} \end{cases}$

□

Moreover, we can show that if the first objective (2.9) is ignored and the ceiling and logarithm functions in (2.10) are removed, then the resulting single objective problem is NP-hard.

Proposition 2. *The gene stacking problem defined by* $\min\left\{\sum_j z_{j,k} : (2.11) - (2.16)\right\}$ *is NP-hard.*

Proof. We also reduce the set covering problem to the gene stacking problem. For a given H, we construct an initial breeding population matrix $A = H$ and we set $K = 1$. The resulting gene stacking problem is then equivalent to the set covering problem in the sense that a solution t is optimal to the set covering problem if and only if $z = t$ is optimal to the gene stacking problem. □

2.4 Algorithm

In this section, we first present an exact algorithm for obtaining the Pareto frontier solutions to the multi-objective optimization models (2.9)-(2.16), and then introduce improvement and heuristics to speedup two steps in the algorithm at the price of possibly compromised solution quality.

2.4.1 Exact Algorithm

In a multi-objective optimization problem, optimality of all objectives are usually not achievable at the same time. An important concept for making wise tradeoffs among competing objectives is *Pareto optimality*. A solution is Pareto optimal if and only if it is not dominated by any feasible solution. A solution x is dominated by $\hat{x}$ if $\hat{x}$ is no worse than x in any objectives and strictly better than x in at least one objective. Under this definition, there could exist multiple Pareto optimal solutions, and the collection of these solutions, called Pareto frontier, yields a range of good tradeoff options for specific decision makers to choose from. Any of these solutions can be claimed optimal, depending on specific situations or perspectives.

In the following, we present an exact algorithm for obtaining the Pareto frontier of problem (2.9)-(2.16).

Step 0: Initialize $\mathcal{P} = \emptyset$ as the set of Pareto optimal solutions.

Step 1: Solve max{(2.9): (2.11)-(2.16)}. Let (x^*, y^*, z^*) be an optimal solution and ρ_1 be the optimal objective value.

Step 2: Solve the following integer program

$$\text{Objective (2.10)} \tag{2.18}$$

$$\text{s. t.} \quad \sum_{i,k} w_k \log(1 - r_i) + \sum_{i,j,k} y_{i,j,k} \log\left(\frac{r_i}{1 - r_i}\right) \geq \rho_1 \tag{2.19}$$

$$\text{Constraints (2.11)-(2.16).} \tag{2.20}$$

Let (x_1, y_1, z_1) be an optimal solution and τ_1 be the optimal objective value. Update $\mathcal{P} = \mathcal{P} \cup (x_1, y_1, z_1)$.

Step 3: Solve the integer program min{(2.10): (2.11)-(2.16)}. Let (x^*, y^*, z^*) be an optimal solution and τ_2 be the optimal objective value .

Step 4: Solve the following integer program

$$\begin{aligned} & \text{Objective (2.9)} && (2.21)\\ \text{s. t.} \quad & \sum_j z_{j,k} \leq 2^{\tau_2}, \forall k && (2.22)\\ & \text{Constraints (2.11)-(2.16).} && (2.23) \end{aligned}$$

Let (x_2, y_2, z_2) be an optimal solution. Update $\mathcal{P} = \mathcal{P} \cup (x_2, y_2, z_2)$.

Step 5: If $\tau_1 - 1 \leq \tau_2$, then stop. Otherwise set $\tau_2 = \tau_2 - 1$ and solve the following integer program

$$\begin{aligned} & \text{Objective (2.9)} && (2.24)\\ \text{s. t.} \quad & \sum_j z_{j,k} \leq 2^{\tau_2}, \forall k && (2.25)\\ & \text{Constraints (2.11)-(2.16).} && (2.26) \end{aligned}$$

Let $(\bar{x}, \bar{y}, \bar{z})$ be an optimal solution, update $\mathcal{P} = \mathcal{P} \cup (\bar{x}, \bar{y}, \bar{z})$, and repeat Step 5. □

In this algorithm, Steps 1 and 2 obtain a Pareto optimal solution (x_1, y_1, z_1) that achieves the maximal likelihood of success with fewest generations; Steps 3 and 4 obtain a Pareto optimal solution (x_2, y_2, z_2) that achieves the fewest generations with maximal likelihood of success; and Step 5 is used repeatedly to compute other Pareto optimal solutions in between. When there is only one variation at each locus, this algorithm automatically reduces to one for the single variation case.

2.4.2 Improvement to Step 1

The objective of Step 1 is to obtain ρ_1, the maximally possible likelihood of success to achieve the target population. An obstacle to the efficient implementation of this step is the

choice of K, the upper bound of the number of inbreds in the target population. If the value of K is selected too small, then it could eliminate the true optimal solution; if it is selected too large, then it would make the dimension of the integer program max{(2.9): (2.11)-(2.16)} too high to be solved efficiently. Here we present a different integer program that not only solves for ρ_1 with much fewer variables but also computes the optimal value for K endogenously:

$$\max_{x,y,w} \quad \rho = \sum_i w \log(1 - r_i) + \sum_{i,j} y_{i,j} \log\left(\frac{r_i}{1 - r_i}\right) \tag{2.27}$$

$$\text{s. t.} \quad \sum_{j:A_{i,j}=0} x_{i,j} = 0, \forall i \tag{2.28}$$

$$\sum_j x_{i,j} = w, \forall i \tag{2.29}$$

$$y_{i,j} \geq x_{i,j} - x_{i+1,j}, \forall i, j \tag{2.30}$$

$$\sum_{j,k:A_{i,j}=v} x_{i,j} \geq 1, \forall i, v \tag{2.31}$$

$$x_{i,j}, y_{i,j}, w \in \mathbb{Z}, \forall i, j. \tag{2.32}$$

The variables x, y, and w in (2.12)-(2.16) are respectively compressed to x, y, and w in (2.27)-(2.32) by summation in the k dimension. The optimal w in (2.27)-(2.32) yields the optimal value for K. Constraints (2.28), (2.29), (2.30) and (2.31) correspond to (2.11), (2.12), (2.13) and (2.15) respectively. The model (2.27)-(2.32) provides an improved formulation to compute ρ_1 for Step 1 of the algorithm, but it does not specify how each inbred in the target population is created, so it cannot be used in other steps when Objective (2.10) may be concerned.

2.4.3 Heuristics to Step 3

The objective of Step 3 is to obtain τ_2, the fewest possible generations to achieve the target population. We propose to solve the following set covering problem to estimate τ_2:

$$\min\{\zeta = \sum_i z_i : \text{sign}(A)z \geq 1, z \in \mathbb{B}^{n\times 1}\}. \tag{2.33}$$

This integer program has much less variables than the one in Step 3, thus is much easier and faster to solve. Let ζ^* denote the optimal objective value of (2.33), then we can conclude that τ_2 satisfies $\lceil \log_2 \zeta^* \rceil \leq \tau_2 \leq \lceil \log_2(\zeta^* + 1) \rceil$. The lower bound is because no inbred in any

target population could possibly be created using less than ζ^* inbreds from the initial breeding population. The upper bound is due to the fact that there exists a naive stacking strategy that can create a biologically diverse target population of inbreds each using at most $\zeta^* + 1$ initial breeding inbreds. This naive breeding strategy is to use the ζ^* inbreds to create a new genotype with no undesirable alleles, and then cross it with another initial inbred to introduce a new variation of desirable allele. By the same means, all variations of all alleles can be introduced to the target population, one through each target inbred, which can be done in parallel, thus a target population can be created within $\lceil \log_2(\zeta^* + 1) \rceil$ generations. In many cases, we could have $\lceil \log_2 \zeta^* \rceil = \lceil \log_2(\zeta^* + 1) \rceil$, then this approach will yield the exact optimal objective value τ_2.

2.5 Computational Experiments

To demonstrate the effectiveness of the algorithms proposed in Section 4.3, we conduct a group of numerical experiments using MATLAB and CPLEX on a PC with 3.25 GB ram and a 3.00 GHz CPU. Two examples from these experiments are explained in detail in this section. Example 1 is the single variation case as introduced in Section 2.3.2, and Example 2 is the biological diversity case as introduced in Section 2.3.3

Example 1. Consider an initial population of inbreds described by the following matrices A and r:

$$A = \begin{bmatrix}
0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 \\
0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\
0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 \\
1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 \\
1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 1 \\
1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 \\
1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\
0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 \\
1 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 \\
1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 \\
0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\
0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 \\
1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\
1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\
1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1
\end{bmatrix},$$

$$r = \begin{bmatrix} 0.0191 \\ 0.0065 \\ 0.0203 \\ 0.0261 \\ 0.0041 \\ 0.0915 \\ 0.0345 \\ 0.0629 \\ 0.0963 \\ 0.0015 \\ 0.0413 \\ 0.0890 \\ 0.0874 \\ 0.0810 \\ 0.0462 \\ 0.0234 \\ 0.0648 \\ 0.0186 \\ 0.0213 \\ 0.0138 \\ 0.0997 \\ 0.0817 \\ 0.0655 \\ 0.0589 \\ 0.0593 \\ 0.0431 \\ 0.0528 \\ 0.0330 \\ 0.0241 \end{bmatrix}.$$

It took around one second to obtain the Pareto frontier, which consists of three points, as illustrated in Figure 5. The none-zero columns of optimal x matrices for the three Pareto

optimal solutions are given as follows.

$$x_2^* = \begin{bmatrix}
1&0&0&0\\1&0&0&0\\1&0&0&0\\0&0&1&0\\1&0&0&0\\1&0&0&0\\0&1&0&0\\0&1&0&0\\0&0&0&1\\0&1&0&0\\0&0&1&0\\0&0&1&0\\0&0&1&0\\0&0&1&0\\0&1&0&0\\0&0&0&1\\0&0&0&1\\0&0&1&0\\0&0&1&0\\0&0&1&0\\0&0&1&0\\0&0&1&0\\0&0&0&1\\0&1&0&0\\0&1&0&0\\1&0&0&0\\0&1&0&0\\0&1&0&0\\0&0&0&1\\0&0&0&1\\0&0&0&1
\end{bmatrix}, x_3^* = \begin{bmatrix}
1&0&0&0&0&0&0&0\\1&0&0&0&0&0&0&0\\1&0&0&0&0&0&0&0\\0&1&0&0&0&0&0&0\\0&1&0&0&0&0&0&0\\0&1&0&0&0&0&0&0\\0&0&0&1&0&0&0&0\\0&0&0&1&0&0&0&0\\0&0&0&0&0&0&1&0\\0&0&0&0&0&0&1&0\\0&0&0&0&0&0&1&0\\0&0&0&0&0&1&0&0\\0&0&0&0&0&1&0&0\\0&0&0&0&0&1&0&0\\0&0&1&0&0&0&0&0\\0&0&1&0&0&0&0&0\\0&0&1&0&0&0&0&0\\0&0&0&0&0&1&0&0\\0&0&0&0&0&1&0&0\\0&0&0&0&0&1&0&0\\0&0&0&0&0&1&0&0\\0&0&0&0&1&0&0&0\\0&0&0&1&0&0&0&0\\0&0&0&1&0&0&0&0\\0&0&0&0&1&0&0&0\\0&0&0&0&1&0&0&0\\0&0&0&0&1&0&0&0\\0&0&0&0&0&0&0&1\\0&0&0&0&0&0&0&1\\0&0&0&0&0&0&0&1
\end{bmatrix}, x_4^* = \begin{bmatrix}
0&1&0&0&0&0&0&0&0\\0&1&0&0&0&0&0&0&0\\0&1&0&0&0&0&0&0&0\\0&0&1&0&0&0&0&0&0\\0&0&1&0&0&0&0&0&0\\0&0&1&0&0&0&0&0&0\\0&0&0&0&0&0&1&0&0\\0&0&0&0&0&0&1&0&0\\0&0&0&0&0&0&1&0&0\\0&0&0&0&0&0&0&0&1\\0&0&0&0&0&0&0&0&1\\0&0&0&0&0&0&0&1&0\\0&0&0&0&0&0&0&1&0\\0&0&0&0&0&0&0&1&0\\0&0&0&0&0&0&0&1&0\\0&0&0&1&0&0&0&0&0\\0&0&0&1&0&0&0&0&0\\0&0&0&1&0&0&0&0&0\\0&0&0&0&0&0&0&1&0\\0&0&0&0&0&0&0&1&0\\0&0&0&0&0&0&0&1&0\\0&0&0&0&0&0&0&1&0\\0&0&0&0&0&1&0&0&0\\0&0&0&0&1&0&0&0&0\\0&0&0&0&1&0&0&0&0\\0&0&0&0&1&0&0&0&0\\0&0&0&0&1&0&0&0&0\\1&0&0&0&0&0&0&0&0\\1&0&0&0&0&0&0&0&0\\1&0&0&0&0&0&0&0&0
\end{bmatrix}.$$

Here x_2^*, x_3^*, and x_4^* correspond to the three points in Figure 5 with numbers of generations being 2, 3, and 4, respectively. If we use their column numbers to refer to the 30 inbreds in the initial population, then the four non-zero columns in x_2^* are inbreds (2), (20), (26), (30); the eight non-zero columns in x_3^* are inbreds (2), (13), (17), (20), (21), (26), (29), and (30); and the nine non-zero columns in x_4^* are inbreds (1), (2), (13), (17), (19), (21), (24), (26), and (29). These Pareto optimal solutions provide three tradeoff options between likelihood of success ρ

and generation time τ. The fastest approach to creating the ideal inbred takes two generations, but the likelihood of success is as low as $\rho_2 = -43.8$. The approach with the highest likelihood of success $\rho_4 = -28.3$ takes four generations. Perhaps a good compromise is x_3^*, which has a likelihood of success $\rho_3 = -28.8$ and takes three generations. All three solutions could be considered optimal in certain situations.

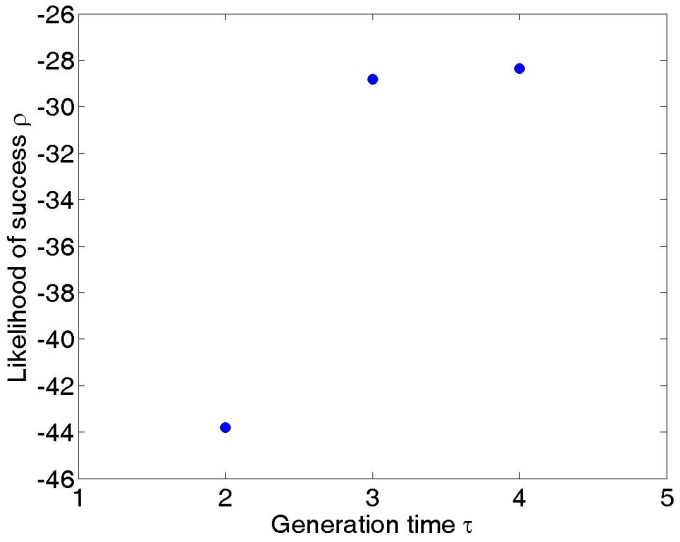

Figure 2.1 Pareto frontier of Example 1.

Example 2. Consider an initial population of inbreds described by the following matrices A and r:

$$A = \begin{bmatrix} 1 & 3 & 0 & 2 & 0 & 0 & 3 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & 0 & 3 & 0 & 2 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 2 & 0 & 0 & 0 & 0 & 0 & 3 \\ 1 & 3 & 0 & 2 & 0 & 0 & 3 & 0 & 1 & 0 & 0 & 0 \\ 0 & 2 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 3 & 1 & 2 \\ 1 & 3 & 0 & 2 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 3 & 0 & 2 & 3 \\ 1 & 0 & 2 & 0 & 1 & 0 & 3 & 0 & 0 & 0 & 2 & 0 \\ 1 & 0 & 0 & 0 & 2 & 3 & 1 & 0 & 0 & 2 & 0 & 0 \\ 0 & 1 & 1 & 1 & 3 & 1 & 1 & 1 & 2 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 & 3 & 2 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 2 & 3 & 1 & 0 & 2 & 0 & 0 & 0 \end{bmatrix}, r = \begin{bmatrix} 0.0770 \\ 0.0350 \\ 0.0662 \\ 0.0416 \\ 0.0842 \\ 0.0833 \\ 0.0256 \\ 0.0613 \\ 0.0582 \\ 0.0541 \\ 0.0870 \end{bmatrix}.$$

The Pareto frontier consists of three solutions, each corresponding to a target population, which contains a varying number of target inbreds. They are illustrated in Figure 6. We use x_k^i to

denote the kth target inbred of the ith Pareto optimal solution. It took 49.1 seconds to compute the first Pareto optimal solution, which contains 10 inbreds in the target population:

$$x_1^1(1,2) = \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \\ 0 & 1 \\ 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 1 & 0 \\ 1 & 0 \end{bmatrix}, x_1^2(1,4) = \begin{bmatrix} 0 & 1 \\ 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \\ 0 & 1 \\ 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 1 & 0 \\ 1 & 0 \end{bmatrix}, x_1^3(1,5) = \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \\ 1 & 0 \\ 1 & 0 \end{bmatrix}, x_1^4(1,6) = \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 1 & 0 \\ 0 & 1 \\ 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \\ 0 & 1 \\ 0 & 1 \end{bmatrix}, x_1^5(1,7) = \begin{bmatrix} 0 & 1 \\ 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \\ 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \\ 0 & 1 \\ 0 & 1 \\ 0 & 1 \end{bmatrix},$$

$$x_1^6(1,8) = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 1 & 0 \\ 1 & 0 \end{bmatrix}, x_1^7(1,9) = \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \\ 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \\ 0 & 1 \end{bmatrix}, x_1^8(1,10) = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \\ 0 & 1 \\ 1 & 0 \end{bmatrix}, x_1^9(1,11) = \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \\ 0 & 1 \\ 1 & 0 \\ 0 & 1 \\ 1 & 0 \end{bmatrix}, x_1^{10}(1,12) = \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 1 & 0 \\ 0 & 1 \\ 1 & 0 \\ 0 & 1 \\ 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 1 & 0 \\ 1 & 0 \end{bmatrix}.$$

The numbers in parentheses indicate the indices of the non-zero columns in the solutions, which correspond to inbreds in the initial population, numbered from 1 to 12 from left to right. Since each target inbred can be created by crossing two from the initial population, the entire target population could be created within one generation. The likelihood of success is as low as $\rho_1 = -151.2$.

It took 21.3 seconds to compute the second Pareto optimal solution, which contains three

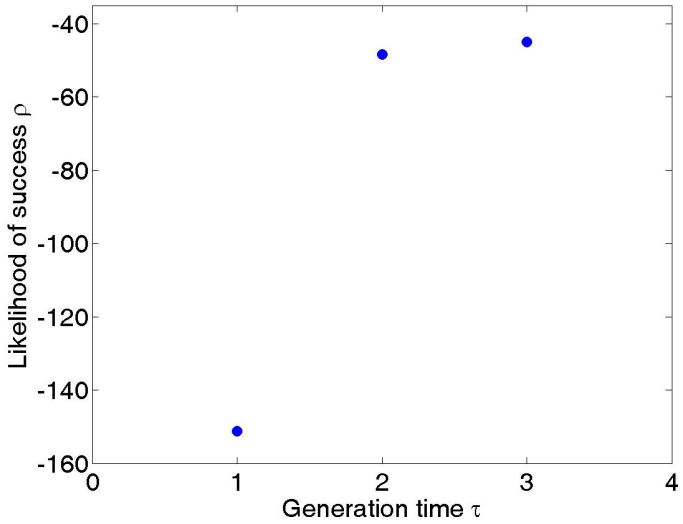

Figure 2.2 Pareto frontier of Example 2.

inbreds in the target population:

$$x_2^1(4,7,8,12) = \begin{bmatrix} 1&0&0&0\\ 0&0&1&0\\ 0&0&0&1\\ 1&0&0&0\\ 1&0&0&0\\ 1&0&0&0\\ 0&0&0&1\\ 0&1&0&0\\ 0&1&0&0\\ 0&1&0&0\\ 0&1&0&0\\ 0&1&0&0 \end{bmatrix}, x_2^2(1,6,9,10) = \begin{bmatrix} 1&0&0&0\\ 0&0&0&1\\ 1&0&0&0\\ 1&0&0&0\\ 0&0&0&1\\ 1&0&0&0\\ 1&0&0&0\\ 1&0&0&0\\ 0&1&0&0\\ 0&0&1&0\\ 0&0&1&0\\ 0&0&1&0 \end{bmatrix}, x_2^3(2,5,6,11) = \begin{bmatrix} 1&0&0&0\\ 1&0&0&0\\ 0&0&1&0\\ 1&0&0&0\\ 1&0&0&0\\ 1&0&0&0\\ 0&0&0&1\\ 0&0&0&1\\ 0&1&0&0\\ 0&1&0&0\\ 0&0&1&0\\ 0&0&1&0 \end{bmatrix}.$$

This solution requires two generations, and the likelihood of success is $\rho_2 = -48.3$.

It took 0.4 second to compute the third Pareto optimal solution, which contains three

inbreds in the target population:

$$x_3^1(2,7,8,12) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}, x_3^2(1,6,9,10) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}, x_3^3(4,5,6,10,11) = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix}.$$

This solution requires three generations, and the likelihood of success is $\rho_3 = -45.0$.

2.6 Conclusions and Discussions

In this paper, we present an optimization approach to stacking desirable genes scattered in multiple inbred lines into one genotype. The issue of biological diversity is also taken into account by preserving desirable alleles of all genetic variations. We formulate this problem as a multi-objective integer programming program, which yields Pareto optimal breeding strategies with respect to two criteria: the likelihood of successfully obtaining the target population of inbreds and the number of generations required to achieve it. Numerical examples demonstrate the feasibility of using the proposed approach to obtain breeding strategies for stacking a smaller amount of genes of a few variations into one target population of inbreds within one minute.

Several limitations of the proposed approach must be noted. First, our approach is based on the assumption that the mapping from genes of interest to target loci is known. The developments in quantitative trait loci (Bernardo and Yu, 2007; Jung et al., 2005; Kebede et al., 2001; Mackay, 2004; Wu et al., 2002) and marker-assisted selection (Bernardo and Charcosset, 2006; Dekkers, 2007; Lande and Thompson, 1990; Lange and Whittaker, 2001; Whittaker, 2001)

provide theoretical basis for such mapping information, yet complete knowledge is not always available or easily accessible. Second, the likelihood of success computed in (2.8) or (2.17) is only an approximate indicator of the probability of success, but does not reflect the exact value. To see the difference, consider the following simple example. Suppose the initial population has two inbreds: $A = \begin{bmatrix} 1 & 0 \\ 1 & 1 \\ 0 & 1 \\ 0 & 1 \end{bmatrix}$, and the recombination probabilities are $r = \begin{bmatrix} 0.01 \\ 0.02 \\ 0.05 \end{bmatrix}$. It is not hard to see that the optimal breeding solution is $x^* = \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \end{bmatrix}$ with a likelihood of success $\rho^* = \log[(1-0.01) \times 0.02 \times (1-0.05)] = -4.0$, since the necessary recombination is easier to occur between the second and third loci than between the first and second ones. However, when the two inbreds are mated, the probability of creating the ideal inbred is not $(1-0.01) \times 0.02 \times (1-0.05)$ but $(1-0.01) \times 0.02 \times (1-0.05) + 0.01 \times (1-0.02) \times (1-0.05)$, since the less likely recombination between the first two loci also contributes to the probability of success. We ignore such contributions in the likelihood calculation in order to avoid summations inside the product of probabilities; otherwise it would void the linearization trick for the objective function, which the tractability of the model is so dependent upon. This modeling inaccuracy has two consequences: (i) the probability of success of the optimal breeding strategy is underestimated to some extent, and (ii) there may exist certain breeding strategies whose computed likelihoods of success are much lower then the actual probabilities thus are mistakenly determined non-optimal by the algorithm. The third limitation is that our model assumes no interference between adjacent loci pairs, which may not always be the case in practice.

Despite the limitations, our approach makes a distinct contribution to the animal and plant breeding practices and theoretical studies on the genetic architecture of complex traits. To our best knowledge, this is the first study that formulates the gene stacking problem as a

multi-objective optimization model. This approach takes advantage of the state-of-the-art algorithms and computer solvers that can efficiently search the entire exponentially exploding (Servin et al., 2004) solution space and provide solutions that can be mathematically proved to be optimal. The flexibility to address biological diversity in the model also extends the model's applicability to a wider range of situations, in which all alleles cannot be simply differentiated as desirable or undesirable but also exhibit distinctions in their desirability. Using results from complexity theory, we prove that the gene stacking problem is NP-hard, which can be non-rigorously interpreted as extremely unlikely to be polynomially solvable. Recently emerging results (Deolalikar, 2010) in complexity theory have presented unconfirmed proofs for the non-existence of polynomial algorithms for NP-hard problems.

It is important to recognize that, although solutions from the optimization models provide insightful information, they are merely tools to enable animal and plant breeders to make better informed decisions when faced with overwhelming amounts of "omics" data for multiple breeding objectives involving complex traits. Consider the same example two paragraphs above. Even though the optimal breeding strategy is given in x^*, it does not address all concerns such as the following two: (i) optimal planting arrangement for founding inbreds to assure successful matings and (ii) what to do if the ideal inbred does not result after the first generation by whatever chance and for whatever reason. To address concern (i), it would require a separate study on the tradeoff between the breeding cost and the probability of success, and it is ultimately individual breeder's call. As for concern (ii), suppose the best child produced after the first generation is $\begin{bmatrix} 1 \\ 1 \\ 1 \\ 0 \end{bmatrix}$ rather than the ideal column of ones, then one could certainly apply our approach to the updated initial population $\tilde{A} = \begin{bmatrix} 1 & 0 & 1 \\ 1 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 1 & 0 \end{bmatrix}$ and obtain another optimal

breeding strategy. Although such dynamic decision making is out of the scope of this study, it could be a promising direction for future research.

CHAPTER 3. AN EXACT ALGORITHM FOR THE BILEVEL MIXED INTEGER LINEAR PROGRAMMING PROBLEM UNDER THREE SIMPLIFYING ASSUMPTIONS

A paper submitted to *Computers and Operations Research*, under review.

Pan Xu, Lizhi Wang, Shan Jin

3.1 Abstract

We present an exact algorithm for the bilevel mixed integer linear programming (BMILP) problem under three simplifying assumptions. Although BMILP has been studied for decades and widely applied to various real world problems, there are only a handful of algorithms. Compared to these existing ones, our new algorithm relies on weaker assumptions, explicitly considers infinite optimal, infeasible, and unbounded cases, and is proved to terminate infinitely with the correct output. We report results of our computational experiments on a small library of BMILP test instances, which we have created and made publicly available online.

3.2 Introduction

We present an exact algorithm for the bilevel mixed integer linear programming (BMILP) problem under three simplifying assumptions. A general BMILP is defined as follows:

$$\max_{x,y} \quad \zeta_1 = c^\top x + d_1^\top y \tag{3.1}$$

$$\text{s. t.} \quad A_1 x + B_1 y \leq b_1 \tag{3.2}$$

$$y \in \text{argmax}_{\tilde{y}}\{\zeta_2 = d_2^\top \tilde{y} : A_2 x + B_2 \tilde{y} \leq b_2; \tilde{y} \geq 0; \tilde{y}_J \in \mathbb{Z}^{|J|}\} \tag{3.3}$$

$$x \geq 0; x_I \in \mathbb{Z}^{|I|}, \tag{3.4}$$

where $A_1 \in \mathbb{R}^{m_1 \times n_1}$, $A_2 \in \mathbb{R}^{m_2 \times n_1}$, $B_1 \in \mathbb{R}^{m_1 \times n_2}$, $B_2 \in \mathbb{R}^{m_2 \times n_2}$, $b_1 \in \mathbb{R}^{m_1 \times 1}$, $b_2 \in \mathbb{R}^{m_2 \times 1}$, $c \in \mathbb{R}^{n_1 \times 1}$, $d_1 \in \mathbb{R}^{n_2 \times 1}$, $d_2 \in \mathbb{R}^{n_2 \times 1}$, $I \subseteq \{1, ..., n_1\}$, and $J \subseteq \{1, ..., n_2\}$. The three simplifying assumptions that our algorithm relies upon are:

Assumption 1: $I = \{1, ..., n_1\}$.

Assumption 2: $\exists X \in \mathbb{R}^{n_1}, x \leq X$.

Assumption 3: $A_2 \in \mathbb{Z}^{m_2 \times n_1}$.

Assumptions 1 and 2 mean, respectively, that all x variables are integral and bounded, and Assumption 3 means that the A_2 matrix is integral. Under these assumptions, the BMILP becomes:

$$\max_{x,y}\{\text{Objective (4.1)} : \text{Constraints (4.2)-(4.3)}; 0 \leq x \leq X; x \in \mathbb{Z}^{n_1}\}. \tag{3.5}$$

The BMILP model we try to solve belongs to the category of bilevel optimization, which has been studied for decades. Pioneers of bilevel optimization models include Bracken and McGill (1973, 1974, 1978), Aiyoshi and Shimizu (1981, 1984), Bard and Falk (1982), and Candler and Townsley (1982), among others. Most early studies (Ben-Ayed, 1993; Ben-Ayed and Blair, 1990; Bialas and Karwan, 1984; Hansen et al., 1992; Wen and Hsu, 1991) focused on the simpler case of bilevel linear programs. Since the 1990s, there has been increased attention on more complex models with either nonlinear terms (Bard and Moore, 1990; Bard, 1988; Edmunds and Bard, 1991, 1992) or discrete decision variables (Bard and Moore, 1992; DeNegre and Ralphs, 2009; Moore and Bard, 1990; Vicente et al., 1996). Comprehensive reviews of existing bilevel optimization algorithms and applications can be found in Colson et al. (2007); Vicente and Calamai (1994).

Bilevel optimization models have been applied to solve a variety of real world problems, in which the hierarchical and competitive structure of decision making widely exists. These applications include revenue management (Cote et al., 2003), network design (Constantin and Florian, 1995), national security (Salmeron et al., 2004), network interdiction (Lim and Smith, 2007; Morton et al., 2007; Yao et al., 2007), national agriculture planning (Fortuny-Amat and

McCarl, 1981), decentralized management of multidivisional firms (Bard, 1983), etc.

Despite its broad applicability, BMILP is intrinsically hard to solve, both theoretically and computationally. This can be epitomized by the simple two-dimensional example from Köppe et al. (2009):

Example 2.

$$\begin{aligned} \sup_{x,y} \quad & \zeta_1 = -x + y \\ \text{s.t.} \quad & 0 \leq x \leq 1 \\ & y \in \text{argmax}_{\tilde{y}} \{\zeta_2 = -\tilde{y} : 0 \leq \tilde{y} \leq 1; \tilde{y} \geq x; \tilde{y} \in \mathbb{Z}\}. \end{aligned}$$

It can be easily checked that: (i) the bilevel feasible region is $\{(0,0)\} \cup ((0,1] \times \{1\})$, which is non-convex and non-continuous, (ii) the supremum of ζ_1 is 1, but it is not attainable, and (iii) the optimal objective value of the continuous relaxation, which is 0, fails to provide an upper bound for ζ_1.

Perhaps due to these challenges, there has been insufficient attention and progress on theoretical and algorithmic development. We were able to find only a handful of BMILP algorithms. In their seminal work (Moore and Bard, 1990), Moore and Bard presented perhaps the first branch-and-bound algorithms for BMILP under the assumption that $B_1 = 0^{m_1 \times n_2}$. The nature of the algorithms is heuristic, which does not guarantee their correctness or finite termination. However, they proved that the algorithms find the optimal solution if either $I = \{1, ..., n_1\}$ or $J = \emptyset$. The case of unbounded BMILP is not discussed. DeNegre and Ralphs (2009) invented another algorithm under the assumptions that $I = \{1, ..., n_1\}$, $J = \{1, ..., n_2\}$, and $B_1 = 0^{m_1 \times n_2}$. No proof of correctness and finite termination is provided, nor is the unbounded case discussed. Relevant literature also includes Köppe et al. (2009) and Dempe (2001). Köppe et al. (2009) is among the first to discover that, if the upper level contains continuous variables and the lower level discrete ones, then the supremum of the BMILP may not be attainable even if it finitely exists. However, the paper only proved the existence of an algorithm that solves

BMILP in polynomial time when n_2 is fixed, referring to parametric integer programming approaches (Eisenbrand and Shmonin, 2008), but no details of the algorithm were provided. The algorithm proposed by Dempe (2001), although referred to as a discrete bilevel programming algorithm, solves a different model, which is more similar to what is referred to as the inverse mixed integer linear programming problem in Wang (2009).

The algorithm that we present here differs from existing ones in the following four aspects. First, we allow for nonzero B_1 matrix. This significantly raises the difficulty of the problem, since obtaining a bilevel feasible solution is no longer straightforward. Nevertheless, allowing for nonzero B_1 could be practically imperative, because in many real world situations the consequences of the lower level's decisions must be taken into account by the upper level as explicit constraints. Second, we allow the lower level to have both continuous and discrete variables. Third, we explicitly consider all possible outcomes of a BMILP, be it infeasible, unbounded, or finite optimal. Fourth, we prove that our algorithm will finitely terminate with the correct output. The three simplifying assumptions are necessary for our algorithm. The first one is to avoid the case of unattainable supremum, the second one is to ensure finite termination, and the third one to avoid rounding errors.

The rest of the paper is organized as follows. We present and interpret the algorithm step by step in Section 4.2, and report results from computational experiments in Section 4.3. Concluding remarks are made in Section 4.4.

3.3 An Exact BMILP Algorithm

3.3.1 Algorithm

In this section, we present an algorithm, referred to as $\text{Alg}^{\text{BMILP}}$, which takes A_1, A_2, b_1, b_2, c, d_1, d_2, and J as input data and outputs the global optimal solution (x^*, y^*, ζ^*) to the BMILP (3.5). The notations of $\zeta^* = -\infty$ and $\zeta^* = \infty$ are used for infeasible and unbounded cases, respectively.

$(x^*, y^*, \zeta^*) = \text{Alg}^{\text{BMILP}}(A_1, A_2, b_1, b_2, c, d_1, d_2, J)$

Step 0: Initialize $x^* = \emptyset, y^* = \emptyset, \zeta^* = -\infty$.

Step 1: Solve the high point problem (4.5)-(4.8).

$$\max_{x,y} \quad \zeta_1 = c^\top x + d_1^\top y \tag{3.6}$$

$$\text{s. t.} \quad A_1 x + B_1 y \leq b_1 \tag{3.7}$$

$$A_2 x + B_2 y \leq b_2 \tag{3.8}$$

$$0 \leq x \leq X, x \in \mathbb{Z}^{n_1} \tag{3.9}$$

$$y \geq 0, y_J \in \mathbb{Z}^{|J|}. \tag{3.10}$$

1(a): If (4.5)-(4.8) is infeasible, then return: (3.5) is infeasible.

1(b): If (4.5)-(4.8) is unbounded, then go to Step 4.

1(c): Let (x^0, y^0) be an optimal solution to (4.5)-(4.8). If $c^\top x^0 + d_1^\top y^0 \leq \zeta^*$, then return: (3.5) is pruned.

1(d): Go to Step 2.

Step 2: Solve the lower level MILP (3.11)-(3.13) with the x^0 from Step 1 as a parameter.

$$\max_{y} \quad \zeta_2 = d_2^\top y \tag{3.11}$$

$$\text{s. t.} \quad A_2 x^0 + B_2 y \leq b_2 \tag{3.12}$$

$$y \geq 0, y_J \in \mathbb{Z}^{|J|}. \tag{3.13}$$

2(a): If (3.11)-(3.13) is unbounded, then return: BMILP (3.5) is infeasible.

2(b): Let $\hat{y}$ be an optimal solution to (3.11)-(3.13). If $d_2^\top y^0 \geq d_2^\top \hat{y}$, then return: (x^0, y^0) is optimal to (3.5).

2(c): If $A_1 x^0 + B_1 \hat{y} \leq b_1$ and $c^\top x^0 + d_1^\top \hat{y} > \zeta^*$, then update $(x^* = x^0, y^* = \hat{y}, \zeta^* = c^\top x^0 + d_1^\top \hat{y})$. In either case, go to Step 3.

Step 3: Create $m_2 + 1$ new BMILP branches, recursively solve these using $\text{Alg}^{\text{BMILP}}$ from Step 1, and return the global output (x^*, y^*, ζ^*). Each of these new BMILP branches is the original BMILP (3.5) with additional constraints to (4.2). For $i = 1, ..., m_2$, the additional constraints for the ith BMILP branch are:

$$(A_2 x + B_2 \hat{y})_j \leq (b_2)_j, \forall j = 1, ..., i-1, \text{ and} \tag{3.14}$$

$$(A_2 x + B_2 \hat{y})_i \geq (b_2)_i + \epsilon_i. \tag{3.15}$$

Here, ϵ_i is set to be $\epsilon_i = \lfloor (b_2 - B_2 \hat{y})_i \rfloor + 1 - (b_2 - B_2 \hat{y})_i$, and the subscript i or j refers to the ith or jth row of the corresponding vector. The additional constraints for the $(m_2 + 1)$st BMILP branch are:

$$A_2 x + B_2 \hat{y} \leq b_2, \text{ and} \tag{3.16}$$

$$d_2^\top y \geq d_2^\top \hat{y}. \tag{3.17}$$

In all of these Constraints (3.14)-(3.17), the $\hat{y}$ from Step 2 is used as a parameter.

Step 4: Recursively use $\text{Alg}^{\text{BMILP}}$ to solve the following BMILP.

$$\max_{x,y} \{0 : \text{Constraints (4.2)-(4.3)}; 0 \leq x \leq X; x \in \mathbb{Z}^{n_1}\}. \tag{3.18}$$

4(a): If (3.18) is infeasible, then return: BMILP (3.5) is infeasible.

4(b): Go to Step 5.

Step 5: Solve the following MILP, and let ζ be the optimal objective value.

$$\max_{\Delta y} \quad d_2^\top \Delta y \tag{3.19}$$

$$\text{s.t.} \quad d_1^\top \Delta y \geq 1 \tag{3.20}$$

$$B_1 \Delta y \leq 0 \tag{3.21}$$

$$B_2 \Delta y \leq 0 \tag{3.22}$$

$$\Delta y \geq 0, \Delta y_J \in \mathbb{Z}^{|J|}. \tag{3.23}$$

5(a): If $\zeta = 0$, then stop: BMILP (3.5) is unbounded.

5(b): If $\zeta < 0$, then recursively use $\text{Alg}^{\text{BMILP}}$ to solve the BMILP (3.5) with an additional constraint $c^\top x + d_1^\top y \leq M$ added to Constraint (4.2). Here M is a large finite number.

This algorithm is diagrammed in Figure 4.2.

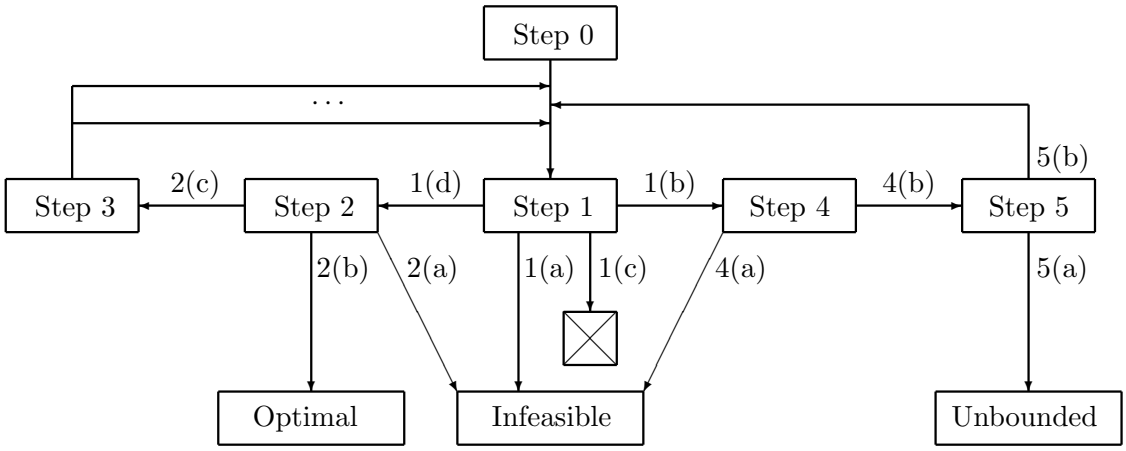

Figure 3.1 Diagram of $\text{Alg}^{\text{BMILP}}$.

3.3.2 Discussion

In this section, we explain the motivation of $\text{Alg}^{\text{BMILP}}$, interpret all the steps, and then prove its correctness and finite termination properties.

$\text{Alg}^{\text{BMILP}}$ is motivated by the following three observations, for which we define (x^*, y^*) and (x^0, y^0) as optimal solutions to the BMILP (3.5) and the high point problem (4.5)-(4.8), respectively.

Observation 1 (Moore and Bard, 1990) The high point problem provides an upper bound for BMILP: $c^\top x^* + d_1^\top y^* \leq c^\top x^0 + d_1^\top y^0$.

Observation 2 If y^0 is optimal to the lower level MILP (3.11)-(3.13), then (x^0, y^0) is optimal to the BMILP (3.5).

Observation 3 For any $\hat{y} \in \mathbb{R}^{n_2}$ with $\hat{y}_J \geq 0$ and $\hat{y}_J \in \mathbb{Z}^{|J|}$, we have either $d_2^\top y^* \geq d_2^\top \hat{y}$ or $(A_2 x^* + B_2 \hat{y})_i > (b_2)_i, \exists i$.

We interpret $\text{Alg}^{\text{BMILP}}$ step by step as follows.

Interpretation – Step 0: We set the initial value for output variables.

Interpretation – Step 1: The high point problem (4.5)-(4.8) can be infeasible, unbounded, or finite optimal.

1(a): If (4.5)-(4.8) is infeasible, then (3.5) is infeasible.

1(b): If (4.5)-(4.8) is unbounded, then we need to examine further in Step 4, because (3.5) can be infeasible, unbounded or finite optimal, as demonstrated in the following BMILP example with $d_2 = 1$, $d_2 = 0$, or $d_2 = -1$, respectively.

Example 3.

$$\begin{aligned} \max_{x,y} \quad & \zeta_1 = x + y \\ \text{s. t.} \quad & 0 \leq x \leq 2, x \in \mathbb{Z} \\ & y \in \text{argmax}_{\tilde{y}} \{\zeta_2 = d_2^\top \tilde{y} : \tilde{y} \geq 1; \tilde{y} \in \mathbb{Z}\}. \end{aligned}$$

1(c): If (4.5)-(4.8) has an optimal solution with $c^\top x^0 + d_1^\top y^0 \leq \zeta^*$, then by Observation 1, (3.5) can be pruned. We emphasize that when all nodes of the branch-and-bound tree are pruned, the original BMILP will be pronounced infeasible.

1(d): If (4.5)-(4.8) has an optimal solution with $c^\top x^0 + d_1^\top y^0 > \zeta^*$, then we need to examine further in Step 2.

Interpretation – Step 2: Since (x^0, y^0) is feasible to (4.5)-(4.8), the lower level MILP (3.11)-(3.13) can only be unbounded or finite optimal.

2(a): If (3.11)-(3.13) is unbounded, then (3.5) is infeasible. This is because (3.11)-(3.13) must possess an extreme ray Δy, which satisfies $\Delta y \geq 0$, $\Delta y_J \in \mathbb{Z}^{|J|}$, $B_2 \Delta y \leq 0$, and $d_2^\top \Delta y > 0$. As a result, for whatever x^0, (3.11)-(3.13) must not possess a finite optimal finite optimal solution, which implies that (4.1)-(4.4) is infeasible.

2(b) If $d_2^\top y^0 \geq d_2^\top \hat{y}$, by Observation 2, (x^0, y^0) is optimal to (3.5).

2(c) If $d_2^\top y^0 < d_2^\top \hat{y}$, we check whether or not the incumbent solution (x^*, y^*) can be updated by $(x^0, \hat{y})$. In either case, we start the branch-and-bound process in Step 3.

Interpretation – Step 3: We create the $m_2 + 1$ branches based on the feasibility of Constraint (3.16). The first m_2 branches are m_2 exclusive and exhaustive ways to violate this constraint. Here the parameter ϵ_i is determined using the assumption that both A_2 and x are integral. The $(m_2 + 1)$st branch satisfies Constraint (3.16), and we also impose (3.17) due to Observation 3. As such, Step 3 eliminates a region that contains (x^0, y^0) and partitions the remaining region into $m_2 + 1$ branches, one of which contains the optimal solutions to (3.5), if it exists. As all active nodes are solved, the incumbent solution (x^*, y^*) is updated globally, which becomes the final output.

Interpretation – Step 4: The BMILP (3.18) can only be infeasible or finite optimal.

4(a): If (3.18) is infeasible, then (3.5) is infeasible.

4(b): If (3.18) is optimal, then we need to examine further in Step 5 since (3.5) can be unbounded or finite optimal, as demonstrated in Example 3.

Interpretation – Step 5: We first show that in Step 5, the MILP (3.19)-(3.23) always has a feasible solution.

Lemma 1. *If (4.5)-(4.8) is unbounded, then (3.19)-(3.23) has a feasible solution.*

Proof. If (4.5)-(4.8) is unbounded, then there exists an extreme ray $(\Delta x, \Delta y)$ such that

$$c^\top \Delta x + d_1^\top \Delta y \geq 1 \tag{3.24}$$

$$A_1 \Delta x + B_1 \Delta y \leq 0 \tag{3.25}$$

$$A_2 \Delta x + B_2 \Delta y \leq 0 \tag{3.26}$$

$$\Delta x = 0, \Delta y \geq 0, \Delta y_J \in \mathbb{Z}. \tag{3.27}$$

These conditions can be further reduced to (3.20)-(3.23). □

Next, we show that since a feasible solution to the BMILP (3.5) has been found in Step 4, the optimal objective value of (3.19)-(3.23) must be non-positive.

Lemma 2. *If (3.19)-(3.23) has a feasible solution with $d_2^\top \Delta y > 0$, then (3.5) is infeasible.*

Proof. If Δy satisfies (3.20)-(3.23) with $d_2^\top \Delta y > 0$, then Δy is also an extreme ray of (3.11)-(3.13), which makes (3.5) infeasible. □

5(a): We now show that if the optimal objective value of (3.19)-(3.23) is zero, then the BMILP (3.5) must be unbounded.

Lemma 3. *If (3.5) has a bilevel feasible solution and the optimal objective value of (3.19)-(3.23) is $d_2^\top \Delta y^* = 0$, then (3.5) is unbounded.*

Proof. Let (x^0, y^0) be bilevel feasible to (3.5) and Δy^* optimal to (3.19)-(3.23). Then it is easy to check that $(x^0, y^0 + \lambda \Delta y^*)$ is bilevel feasible for feasible for all $\lambda \in \mathbb{Z}^+$. Since $d_1^\top \Delta y^* \geq 1$, we conclude that (4.1)-(4.4) is unbounded. □

5(b): If the optimal objective value of (3.19)-(3.23) is negative, we need to further examine because the BMILP (3.5) has a finite optimal solution.

Lemma 4. *If (3.5) has a bilevel feasible solution and the optimal objective value of (3.19)-(3.23) is $d_2^\top \Delta y^* < 0$, then (3.5) has a finite optimal solution.*

Proof. We prove by contradiction. Suppose (3.5) is unbounded, then there must exist a bilevel feasible solution, say (x^0, y^0), and an extreme ray Δy such that $d_1^\top \Delta y \geq 1$ and $(x^0, y^0 + \lambda \Delta y)$ is bilevel feasible for all $\lambda \in \mathbb{Z}^+$. It is easy to check that Δy is feasible to (3.20)-(3.23) with $d_2^\top \Delta y = 0$, which contradicts the fact that $d_2^\top \Delta y^* < 0$. □

The parameter M in Step 5(b) can be obtained using the following procedure. (I) Initialize M with any large positive number. (II) Solve the BMILP (3.5) with additional constraints $M \leq c^\top x + d_1^\top y \leq 2M$ added to Constraint (4.2). If the BMILP is infeasible, then stop. Otherwise set $M = 2M$ and repeat (II).

In the following, we establish the correctness and finite termination of $\text{Alg}^{\text{BMILP}}$.

Theorem 1. *Alg^{BMILP} terminates finitely with the correct output for (3.5).*

Proof. Since a BMILP may be finite optimal, unbounded, or infeasible, we prove for these three possibilities separately.

For a BMILP (3.5) with a finite optimal solution, there are two possible outlets. The more obvious but less frequently observed one is 2(b), under which case the finite optimality claim is guaranteed by Observation 2. The other outlet is a combination of 2(c) and 1(c). The former keeps updating the incumbent solution, which will be claimed optimal after all nodes of the branch-and-bound tree have been pruned in the latter case. We need to prove two more points: (I) Step 3 never eliminates the optimal solution, and (II) the branch-and-bound tree has a finite

depth.

Point (I) can be proved using Observation 3, which means that when (3.16) is satisfied, the optimal solution to (3.5) must also satisfy (3.17). Since the first m_2 branches are m_2 exclusive and exhaustive ways to violate (3.16) and the (m_2+1)st branch satisfies (3.16), the optimal solution must be contained in one of these branches.

To prove Point (II), consider any given path from level 0 to level k (finite or infinite) of the branch-and-bound tree. Let $\{(x^0,y^0),(x^1,y^1),...,(x^k,y^k)\}$ be the optimal solutions to (4.5)-(4.8) of the path. We can show that at most two of these solutions can share the same x. Suppose $x^i = x^j$ for some $i < j$, then x^i must have survived the eliminating cut (3.15) generated in the ith iteration by hiding in the (m_2+1)st branch. However, when $x^j = x^i$ becomes the optimal solution to (4.5)-(4.8) of level j, y^j needs to satisfy cut (3.17) generated from level i, which qualifies (x^j,y^j) to be optimal to (3.5) in Step 2(b), thus the algorithm will terminate and the depth of this path is $k = j$. Since x can only take integer values in a bounded region, the number of different x values is finite, so is the depth of the branch-and-bound tree.

For an unbounded BMILP (3.5), it is easy to see that (3.5) must directly follow 1(b) and 4(b) to Step 5. By Lemmas 1-4, 5(a) is the only possible outcome.

For an infeasible BMILP (3.5), it is easy to see that (4.1)-(4.4) will not terminate via 2(b) or 5(a). It is also obvious that in the case of 1(a), 2(a), or 4(a), (4.1)-(4.4) is indeed infeasible. The only other possible outlet of the algorithm is 1(c), when all nodes in the branch-and-bound tree are pruned, which means that (4.1)-(4.4) is infeasible. □

3.4 Computational Experiments

In this section, we report our computational experiences with $\text{Alg}^{\text{BMILP}}$. This algorithm is implemented in Matlab using TOMLAB/CPLEX as the MILP solver. Computational experiments are executed on a desktop computer with a 3 GB ram and a 2.4 GHz CPU. We

created 11 BMILP test instances, with 8 from MIPLIB (Achterberg et al., 2006), one from Yao et al. (2007), and two from construction. For the 8 MIPLIB instances, we used the MILP data for B_2, d_2, and J for the lower level, and randomly created other parameters. If an instance appeared trivial (e.g., terminated in one iteration or less than a second), we replaced it with another randomly created set of parameters. The instance "Yao" is a modified version from a case study in Yao et al. (2007), which formulated a power system network defense problem as a trilevel optimization problem and designed a heuristic algorithm to solve it. We reformulated the problem as an equivalent BMILP and were able to obtain the global optimal solution. We also manually generated two instances with one being infeasible and the other unbounded. Computational results are summarized in Table 4.1. All of these test instances as well as our solutions are posted online at http://lzwang.public.iastate.edu/bmilplib.html.

Table 3.1 Computational performance of $\text{Alg}^{\text{BMILP}}$.

Instance	m_1, n_1, m_2, n_2	ζ^*	CPU time	# Nodes visited
pk1	10, 10, 45, 86	1,027	4m-9s	28,886
marshare1	100, 100, 6, 62	844	2m-52s	8
marshare2	100, 100, 7, 74	1,213	3m-28s	9
pp08a	10, 10, 136, 240	1,532	2m-30s	4,001
noswot	10, 10, 364, 128	4,248	47s	3,406
mas74	10, 10, 13, 151	741	6m-21s	1,329
mas76	10, 10, 12, 151	167	1h-28m-32s	14
misc07	10, 10, 424, 260	1,327	32m-14s	42,157
Yao	47, 23, 896, 294	–1,854,313	8m-27s	55
inf5	5, 5, 5, 5	$-\infty$	22m-32s	211,111
unbd10	10, 10, 12, 152	∞	1h-38m-24s	$1 + 1^*$

* It took one node to find an extreme ray and another one a bilevel feasible solution.

3.5 Conclusion

In this paper, we present an exact algorithm for the BMILP problem under three simplifying assumptions. Our study makes the following contributions to the existing literature. First, $\text{Alg}^{\text{BMILP}}$ relies on weaker simplifying assumptions than existing algorithms. Second, our algo-

rithm explicitly considers all possible outcomes of a BMILP, including infeasible, unbounded, and finite optimal. Third, we provide a proof of the correctness and finite termination of $\text{Alg}^{\text{BMILP}}$. Fourth, we demonstrate the efficiency of our algorithm by reporting our computational results on small- to medium-sized instances. Last but not least, we have created a library of BMILP test instances for benchmark and comparison of future algorithms.

Our future work is to relax the first two simplifying assumptions that $\text{Alg}^{\text{BMILP}}$ is dependent upon. To allow for both continuous and discrete variables on the upper level, the algorithm should be able to obtain an ϵ-optimal solution (Köppe et al., 2009) when the supremum is not attainable. It may require a completely new algorithm to relax Assumption 2, without which $\text{Alg}^{\text{BMILP}}$ may fail to terminate. Assumption 3 can be easily relaxed to be $A_2 \in \mathbb{Q}^{m_2 \times n_1}$, which is a common assumption made by most literature on discrete optimization. Moreover, the formulation (4.1)-(4.4) implicitly makes an optimistic assumption that the upper level can choose the lower level's optimal solution when multiple ones exist. It would be interesting to extend the model and algorithm to a pessimistic setting by replacing Constraint (4.3) with

$$y \in \text{argmin}\{d_1^\top y : \text{Constraint (4.3)}\},$$

which means that the lower level will choose the worst optimal solution for the upper level when multiple ones exist.

We conclude with a counterexample to demonstrate the necessity of Assumption 2 for $\text{Alg}^{\text{BMILP}}$.

Example 4.

$$\begin{aligned} \max_{x,y} \quad & \zeta_1 = x - y \\ \text{s.t.} \quad & x \geq 1, y \geq 2x, x \in \mathbb{Z} \\ & y \in \text{argmax}_{\tilde{y}}\{\zeta_2 = -\tilde{y} : \tilde{y} \geq x; \tilde{y} \geq 0; \tilde{y} \in \mathbb{Z}\}. \end{aligned}$$

As illustrated in Figure 3.2, the solid dots are feasible solutions to the high point problems, whereas the circles are the optimal solutions to the lower level, which all lie on the 45° line.

Obviously, this BMILP is infeasible, but $\text{Alg}^{\text{BMILP}}$ would solve a series of high point problems with the optimal solutions being $\{(1,2),(2,4),(3,6),...\}$, none of which would be optimal to the lower level, thus the algorithm would go on without termination.

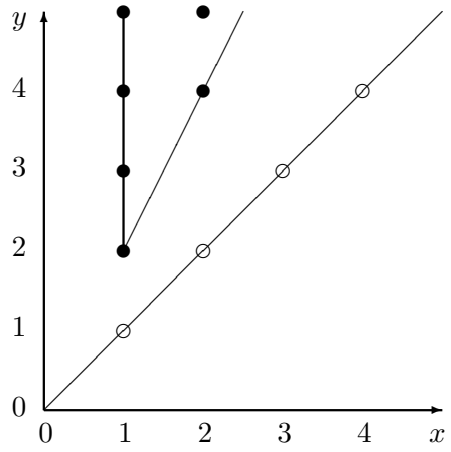

Figure 3.2 Illustration of Example 3.

CHAPTER 4. THE WATERMELON ALGORITHM FOR THE BILEVEL INTEGER LINEAR PROGRAMMING PROBLEM

4.1 Introduction

We present the so-called watermelon algorithm for the bilevel integer linear programming (BILP) problem defined as follows.

$$\max_{x,y} \quad \zeta = c^\top x + d_1^\top y \tag{4.1}$$

$$\text{s. t.} \quad A_1 x + B_1 y \leq b_1 \tag{4.2}$$

$$y \in \operatorname{argmax}_{\tilde{y}} \{ d_2^\top \tilde{y} : A_2 x + B_2 \tilde{y} \leq b_2; \tilde{y} \in \mathbb{Z}^{n_2} \} \tag{4.3}$$

$$x \in \mathbb{Z}^{n_1}. \tag{4.4}$$

Here $A_1 \in \mathbb{R}^{m_1 \times n_1}$, $B_1 \in \mathbb{R}^{m_1 \times n_2}$, $b_1 \in \mathbb{R}^{m_1}$, $c \in \mathbb{R}^{n_1}$, $d_1 \in \mathbb{R}^{n_2}$, and $d_2 \in \mathbb{R}^{n_2}$ are real number parameters and, without loss of generality, $A_2 \in \mathbb{Z}^{m_2 \times n_1}$, $B_2 \in \mathbb{Z}^{m_2 \times n_2}$, and $b_2 \in \mathbb{Z}^{m_2 \times 1}$ are assumed to be integral. The upper level's decision variable are x and y, and the latter is also required by Constraint (4.3) to be an optimal solution to the lower level problem.

BILP belongs to a broader class of bilevel discrete optimization models, which has found many applications in a variety of real-world problems, including large steel structure design (Sarma and Adeli, 2001), natural gas cash-out (Dempe et al., 2005), network design (Gao et al., 2005), taxation policy (He et al., 2011; Zhou et al., 2011), network interdiction (Morton et al., 2007; Lim and Smith, 2007), and critical infrastructure defense (Scaparra and Church, 2008; Yao et al., 2007). In these and many other applications, decisions are made in a hierarchical structure, and integer variables are indispensable to describe discrete decisions such as logical decisions, mutually exclusive choices, and fixed costs.

Bilevel discrete optimization problems are notoriously hard to solve, and there are many

pitfalls when standard single-level discrete optimization theories and algorithms are directly applied (DeNegre and Ralphs, 2009; Moore and Bard, 1990; Vicente et al., 1996). For example, if the upper-level contains continuous variables and the lower-level contains integer variables, then an optimal solution may not exist because the supremum of the objective value may be unattainable albeit finite. Such a pathological instance can be found in Köppe et al. (2009). Moreover, if integrality requirements on x and y are relaxed, then the BILP reduces to a bilevel linear program (BLP). However, even if the optimal solution to the BLP relaxation happens to be integral, it may or may not be optimal to BILP. Such a pathological instance can be found in Moore and Bard (1990). In fact, the optimal objective value to the BLP relaxation provides neither an upper bound nor a lower bound for BMILP. For instance, the BLP relaxation may be infeasible when the BILP has an optimal solution. A more useful relaxation is the so-called high point problem (HPP) (Candler and Townsley, 1982; Moore and Bard, 1990):

$$\max_{x,y} \quad c^\top x + d_1^\top y \tag{4.5}$$

$$\text{s. t.} \quad A_1 x + B_1 y \leq b_1 \tag{4.6}$$

$$A_2 x + B_2 y \leq b_2 \tag{4.7}$$

$$x \in \mathbb{Z}^{n_1}, y \in \mathbb{Z}^{n_2}. \tag{4.8}$$

This integer linear program (ILP) removes the lower-level objective function and allows the upper-level to unilaterally determine both x and y; thus, it provides an upper bound for the BILP.

The literature on discrete bilevel optimization algorithms is sparse. Most of it is approximate or heuristic in nature (Bard and Moore, 1992; Moore and Bard, 1990) and/or relies on simplifying assumptions. For example, the bilevel feasible set is assumed to be bounded in Köppe et al. (2009); it is further assumed in DeNegre and Ralphs (2009); Moore and Bard (1990) that not only the bilevel feasible set is nonempty and bounded but also the lower level feasibility region is nonempty for any feasible upper level decision; all lower level decisions are assumed to have finite lower and upper bounds in Gümüş and Floudas (2005).

The watermelon algorithm we present in this paper differs from previous algorithms by having the following four features: (1) solves all BILP instances defined in (4.1)-(4.3) without

additional simplifying assumptions, (2) terminates in a finite number of iterations, (3) correctly identifies unbounded or infeasible instances, and (4) obtains a global optimal solution if one exists. Notice that a BILP defined in (4.1)-(4.3) only has three possible outcomes: infeasible, unbounded, or optimal (having a global optimal solution).

Here we clarify some definitions that will be used in the rest of the paper. The lower level problem (LLP) refers to the following parametric ILP (4.9)-(4.11), denoted as $\mathcal{L}(x)$:

$$\max_{y} \quad d_2^\top y \tag{4.9}$$

$$\text{s.t.} \quad A_2 x + B_2 y \leq b_2 \tag{4.10}$$

$$y \in \mathbb{Z}^{n_2}. \tag{4.11}$$

A solution (x, y) is said to be *upper level feasible* if it satisfies Constraints (4.2) and (4.4). A solution (x, y) is said to be *lower level feasible/optimal* if y is feasible/optimal to $\mathcal{L}(x)$. A solution (x, y) is *HPP feasible* if it satisfies Constraints (4.6)-(4.8). A solution (x, y) is said to be *bilevel feasible* if it satisfies Constraints (4.2)-(4.4); otherwise it is said to be *bilevel infeasible*. The following examples demonstrate the difference between infeasible, unbounded, and optimal BILPs.

Example 5. *The BILP* (4.12)-(4.14) *is infeasible because the lower level is unbounded for any* x*, thus a solution* (x, y) *that satisfies Constraint* (4.14) *does not exist.*

$$\max_{x,y} \quad x \tag{4.12}$$

$$\text{s.t.} \quad x \geq 2, x \in \mathbb{Z} \tag{4.13}$$

$$y \in \text{argmax}_{\tilde{y}}\{\tilde{y} : \tilde{y} \geq x; \tilde{y} \in \mathbb{Z}\}. \tag{4.14}$$

Example 6. *The BILP* (4.15)-(4.17) *is unbounded because for any real number* K*,* $(x = 2, y = \max\{\lceil K \rceil, 2\})$ *is a bilevel feasible solution with* $\zeta(x, y) \geq K$*.*

$$\max_{x,y} \quad \zeta = y \tag{4.15}$$

$$\text{s.t.} \quad x \geq 2, x \in \mathbb{Z} \tag{4.16}$$

$$y \in \text{argmax}_{\tilde{y}}\{0 : \tilde{y} \geq x; \tilde{y} \in \mathbb{Z}\}. \tag{4.17}$$

Example 7. *The BILP* (4.18)-(4.20) *has an optimal solution* $(x^* = 0, y^* = 0)$*, which cannot be dominated by any bilevel feasible solution.*

$$\max_{x,y} \quad y \tag{4.18}$$

$$\text{s. t.} \quad x \geq 0, x \in \mathbb{Z}; \tag{4.19}$$

$$y \in \text{argmax}_{\tilde{y}}\{-\tilde{y} : 0 \leq \tilde{y} \leq 2x; \tilde{y} \in \mathbb{Z}\}. \tag{4.20}$$

We also present a sufficient condition to identify infeasible BILPs.

Lemma 5. *For any* $x \in \mathbb{R}^{n_1}$*, if* $\mathcal{L}(x)$ *is unbounded, then the BILP* (4.1)-(4.4) *is infeasible.*

Proof. If $\mathcal{L}(x)$ is unbounded, then it must possess an extreme ray Δy such that $\Delta y \in \mathbb{Z}^{n_2}$, $B_2 \Delta y \leq 0$, and $d_2^\top \Delta y \geq 1$. As a result, Constraint (4.3) can never be satisfied, thus the BILP (4.1)-(4.4) is infeasible. □

4.2 The Watermelon Algorithm for BILP

In this section we present the watermelon algorithm for BILP, which is named after the fruit due to some interesting analogies we can draw between the solution technique and how one could eat a watermelon. Several definitions are made metaphorically as follows. We define the *watermelon* as the polyhedron $\{(x, y) : A_2 x + B_2 y \leq b_2\}$. As such the facets of the watermelon will be referred to as the *watermelon skin*, and the interior of the watermelon will be called the *watermelon flesh*. *Watermelon seeds* are defined as those solutions that are HPP feasible but bilevel infeasible. The idea of the watermelon algorithm is the following. We first solve the constrained HPP, which is defined as the HPP within the watermelon region. If the constrained HPP solution is bilevel feasible, then it is optimal to the BILP; otherwise it is a watermelon seed. Next we remove the seed from the watermelon and solve the new constrained HPP again. These steps are repeated and a new seed is removed from the watermelon in each iteration until the optimal BILP solution is found.

To accelerate this algorithm, when a constrained HPP solution is found to be a seed, rather than picking the single seed out, we use a scoop to carve out a piece of watermelon flesh that also contains as many surrounding seeds as possible. We define a *scoop* as the parametric

polyhedron $\{(x, y) : A_2x + B_2y \leq b_2 - t\}$. Here $t \in \mathbb{R}_+^{m_2}$ can be interpreted as the distance between the scoop and the watermelon skin. Notice that the scoop has the same shape with the watermelon. When a scoop is designed to carve out a seed, there are three criteria: (i) the scoop should contain as many watermelon seeds as possible; (ii) the scoop must enclose the target seed (optimal solution to the previous constrained HPP); and (iii) the scoop must not contain any bilevel feasible solution. We use the following parametric integer linear program (ILP) to design this scoop, which will be referred to as *the scoop problem* (SP), denoted as $\mathcal{S}(x^{\mathrm{H}}, y^{\mathrm{H}})$:

$$\min_{\Delta y, t} \quad 1^\top t \tag{4.21}$$

$$\text{s. t.} \quad d_2^\top \Delta y \geq 1 \tag{4.22}$$

$$B_2 \Delta y \leq t \tag{4.23}$$

$$0 \leq t \leq b_2 - A_2 x^{\mathrm{H}} - B_2 y^{\mathrm{H}} \tag{4.24}$$

$$t \in \mathbb{Z}^{m_2}, \Delta y \in \mathbb{Z}^{n_2}. \tag{4.25}$$

Let $(\Delta y^{\mathrm{S}}, t^{\mathrm{S}})$ be an optimal solution to $\mathcal{S}(x^{\mathrm{H}}, y^{\mathrm{H}})$, where $(x^{\mathrm{H}}, y^{\mathrm{H}})$ is the target seed. The objective function (4.21) meets criterion (i). Intuitively, a smaller t means a larger scoop and more seeds. Constraint (4.24) ensures criterion (ii). To show that criterion (iii) is guaranteed, we present the following lemma.

Lemma 6. *If $(\Delta y^S, t^S)$ is an optimal solution to (4.21)-(4.25), then no bilevel feasible solution exists in the scoop defined by $A_2x + B_2y \leq b_2 - t^S$.*

Proof. Prove by contradiction. Suppose (x^0, y^0) is a bilevel feasible solution enclosed in the scoop, then it is easy to check that $(x^0, y^0 + \Delta y^{\mathrm{S}})$ dominates (x^0, y^0) in the lower level problem, which contradicts the assumed lower level optimality of (x^0, y^0). □

We also discuss two more interesting properties of the SP in the following two lemmas.

Lemma 7. *An HPP feasible solution (x^H, y^H) is a watermelon seed if and only if the SP $\mathcal{S}(x^H, y^H)$ has an optimal solution.*

Proof. The "only if" direction: If $(x^{\mathrm{H}}, y^{\mathrm{H}})$ is a watermelon seed, then by definition there must exist a Δy^0 such that $A_2 x^{\mathrm{H}} + B_2(y^{\mathrm{H}} + \Delta y^0) \leq b_2$ and $d_2^\top \Delta y^0 \geq 1$ (due to the integrality assumption on d_2). It is easy to check that $(\Delta y = \Delta y^0, t = \max\{B_2 \Delta y, 0\})$ is a feasible solution to (4.21)-(4.25). Since (4.21)-(4.25) is bounded, it must have an optimal solution.

The "if" direction: Let $(\Delta y^0, t^0)$ be an optimal solution to (4.21)-(4.25). Then it is easy to check that $(x^{\mathrm{H}}, y^{\mathrm{H}} + \Delta y^0)$ dominates $(x^{\mathrm{H}}, y^{\mathrm{H}})$ in the lower level problem, thus the latter is a watermelon seed. □

Lemma 8. *If $(\Delta y = \Delta y^S, t = t^S)$ is an optimal solution to (4.21)-(4.25), so is $(\Delta y = \Delta y^S, t = \max\{B_2 \Delta y^S, 0\})$.*

Proof. First, we show that $(\Delta y = \Delta y^{\mathrm{S}}, t = \max\{B_2 \Delta y^{\mathrm{S}}, 0\})$ is a feasible solution to (4.21)-(4.25). From Constraint (4.23) and $t \geq 0$, we have

$$\max\{B_2 \Delta y^{\mathrm{S}}, 0\} \leq t^{\mathrm{S}}, \tag{4.26}$$

which ensures the feasibility of (4.24). By (4.26), we also have $1^\top \max\{B_2 \Delta y^{\mathrm{S}}, 0\} \leq 1^\top t^{\mathrm{S}}$, which verifies the optimality of $(\Delta y = \Delta y^{\mathrm{S}}, t = \max\{B_2 \Delta y^{\mathrm{S}}, 0\})$. □

After the scoop of flesh is carved out, the remaining part of the watermelon becomes non-convex, because it is inside the larger convex watermelon but outside the smaller convex scoop. To solve the constrained HPP, we propose to first partition the watermelon into $m_2 + 1$ pieces. For $i = 1, ..., m_2$, the ith piece is defined as $\{(x, y) : (A_2 x + B_2 y)_j \leq (b_2 - t^{\mathrm{S}})_j, \forall j = 1, ...i - 1; (b_2 - t^{\mathrm{S}}) + 1 \leq (A_2 x + B_2 y)_i \leq (b_2)_i; (A_2 x + B_2 y)_j \leq (b_2)_j, \forall j = i + 1, ..., m_2\}$. The $(m_2 + 1)$st piece is the scoop of flesh. Then the constrained HPP can be solved separately within the first m_2 pieces in a branch-and-bound framework.

Although the HPP originated from decades ago (Bialas and Karwan, 1984; Candler and Townsley, 1982; Moore and Bard, 1990), it has almost always been assumed to be bounded, which restricts the applicability of previous algorithms to bounded BILPs only. It is easy to see that the HPP of an unbounded BILP must be unbounded. However, the reverse is not true. In fact, the HPPs of Examples 5-7 are all unbounded, although the BILPs have three different outcomes. Therefore, it is necessary for a thorough algorithm to be able to differentiate the

causes of unbounded HPPs. For such purpose, we present the following ILP, which will be referred to as the *blind scoop problem* (BSP):

$$\min_{x,y,\Delta y,t} \quad 1^\top t \tag{4.27}$$

$$\text{s. t.} \quad d_2^\top \Delta y \geq 1 \tag{4.28}$$

$$B_2 \Delta y \leq t \tag{4.29}$$

$$A_1 x + B_1 y \leq b_1 \tag{4.30}$$

$$A_2 x + B_2 y \leq b_2 - t \tag{4.31}$$

$$t \in \mathbb{Z}_+^{m_2}, x \in \mathbb{Z}^{n_1}, y \in \mathbb{Z}^{n_2}, \Delta y \in \mathbb{Z}^{n_2}. \tag{4.32}$$

Similar to the SP, the BSP also tries to design a scoop to carve out a piece of watermelon flesh. The difference is, since no optimal solution is provided by the HPP, the BSP needs to search for a watermelon seed by itself, which explains the word "blind" in the name. By Lemma 7, the existence of a seed is a necessary and sufficient condition for the BSP to have an optimal solution. If BSP finds a seed and a scoop to enclose it, then no bilevel feasible solution is admitted to the scoop.

Lemma 9. *If $(x^B, y^B, \Delta y^B, t^B)$ is an optimal solution to (4.27)-(4.32), then no bilevel feasible solution exists in the scoop defined by $A_2 x + B_2 y \leq b_2 - t^B$.*

Proof. Similar to the proof of Lemma 6. □

If the unboundedness of the HPP was caused by these seeds, then the removal of the scoop may make the constrained HPP bounded in the next iteration. If it turns out that the scoop contains the entire watermelon ($t = 0$), then the BILP is apparently infeasible:

Lemma 10. *If $(x^B, y^B, \Delta y^B, t^B = 0)$ is an optimal solution to (4.27)-(4.32), then the BILP (4.1)-(4.4) is infeasible.*

Proof. If $t^{\mathrm{B}} = 0$ is optimal to (4.27)-(4.32), then Δy^{B} becomes an integral extreme ray of the LLP. As such, Constraint (4.3) can never be satisfied, and the BILP (4.1)-(4.4) is therefore infeasible. □

However, as the following lemma explains, if BSP fails to find a seed, then the unboundedness can only be caused by bilevel feasible solutions, thus the BILP is indeed unbounded.

Lemma 11. *If* (4.5)-(4.8) *is unbounded and* (4.27)-(4.32) *infeasible, then the BILP* (4.1)-(4.4) *is unbounded.*

Proof. Assuming that (4.5)-(4.8) is unbounded and (4.27)-(4.32) infeasible, we show that for any real number K, there exists a bilevel feasible solution (x_K, y_K) such that $c^\top x_K + d_1^\top y_K \geq K$.

Since (4.5)-(4.8) is unbounded, for any real number K, there exists a feasible HPP solution $(x^{\mathrm{H}}, y^{\mathrm{H}})$ such that $c^\top x^{\mathrm{H}} + d_1^\top y^{\mathrm{H}} \geq K$. Since (4.27)-(4.32) is infeasible, (4.21)-(4.25) is also infeasible. By Lemma 7, $(x^{\mathrm{H}}, y^{\mathrm{H}})$ is bilevel feasible. □

The following lemma presents a similar result with Lemma 8.

Lemma 12. *If* $(\Delta y = \Delta y^B, t = t^B)$ *is optimal to* (4.27)-(4.32)*, so is* $(\Delta y = \Delta y^B, t = \max\{B_2 \Delta y^B, 0\})$.

Proof. Similar to the proof of Lemma 8. □

To accommodate the branch-and-bound structure of the watermelon algorithm, we define the parametric versions of HPP and BSP below. The parametric high point problem (PHPP), denoted as $\mathcal{H}(l, u)$, is defined as the following ILP.

$$\max_{x,y}\{\text{Objective (4.5)} : \text{Constraints (4.6) and (4.8)}; l \leq A_2 x + B_2 y \leq u\}. \tag{4.33}$$

The parametric blind scoop problem (PBSP), denoted as $\mathcal{B}(l, u)$, is defined as the following ILP.

$$\max_{x,y}\{\text{Objective (4.27)} : \text{Constraints (4.28)-(4.30) and (4.32)}; l \leq A_2 x + B_2 y + t \leq u\}. \tag{4.34}$$

We are now ready to present the watermelon algorithm for BILP, also referred to as $\mathrm{WAlg}^{\mathrm{BILP}}$. $\mathrm{WAlg}^{\mathrm{BILP}}$ takes A_1, A_2, B_1, B_2, b_1, b_2, c, d_1, d_2 as input data and outputs the global optimal solution (x^*, y^*, ζ^*) to the BILP (4.1)-(4.4). The notations of $(x^* = \emptyset, y^* = \emptyset, \zeta^* = -\infty)$ and $(x^* = \emptyset, y^* = \emptyset, \zeta^* = +\infty)$ are used for infeasible and unbounded cases, respectively. $\mathrm{WAlg}^{\mathrm{BILP}}$ is diagrammed in Figure 4.2.

$(x^*, y^*, \zeta^*) = \mathrm{WAlg}^{\mathrm{BILP}}(A_1, A_2, B_1, B_2, b_1, b_2, c, d_1, d_2)$

Step 0: Create the root node: $(l^1 = -\infty^{m_2}, u^1 = b_2, z^1 = +\infty)$. A node k is characterized by (l^k, u^k, z^k). The first two parameters feed to the PHPP and PBSP and the third one is an upper bound of its optimal objective value. Initialize $x^* = \emptyset, y^* = \emptyset, \zeta^* = -\infty, N = 1$. Here, N is the number of remaining nodes. Go to Step 1.

Step 1: For all $j \in \{1, ..., N\}$ such that $z^j \leq \zeta^*$ or $l^j \not\leq u^j$, remove node j. Update N as the number of remaining nodes.

1(a): If $N = 0$ and $x^* \neq \emptyset$, then stop. Output (x^*, y^*, ζ^*) as the optimal solution.

1(b): If $N = 0$ and $x^* = \emptyset$, then stop. The BILP (4.1)-(4.4) is infeasible.

1(c): Select a node $k \in \{1, ..., N\}$, set $\hat{l} = l^k$ and $\hat{u} = u^k$, remove this node, reduce N by 1, and go to Step 2.

Step 2: Solve the PHPP $\mathcal{H}(\hat{l}, \hat{u})$.

2(a): If $\mathcal{H}(\hat{l}, \hat{u})$ is infeasible, then go to Step 1.

2(b): If $\mathcal{H}(\hat{l}, \hat{u})$ is unbounded, then go to Step 6.

2(c): Let $(x^{\mathrm{H}}, y^{\mathrm{H}})$ be an optimal solution to $\mathcal{H}(\hat{l}, \hat{u})$. If $c^\top x^{\mathrm{H}} + d_1^\top y^{\mathrm{H}} \leq \zeta^*$, then go to Step 1.

2(d): Go to Step 3.

Step 3: Solve the LLP $\mathcal{L}(x^{\mathrm{H}})$.

3(a): If $\mathcal{L}(x^{\mathrm{H}})$ is unbounded, then stop. The BILP (4.1)-(4.4) is infeasible.

3(b): Let y^{L} be an optimal solution to $\mathcal{L}(x^{\mathrm{H}})$. If $d_2^\top y^{\mathrm{H}} \geq d_2^\top y^{\mathrm{L}}$, then update $(x^* = x^{\mathrm{H}}, y^* = y^{\mathrm{H}}, \zeta^* = c^\top x^{\mathrm{H}} + d_1^\top y^{\mathrm{H}})$ and go to Step 1.

3(c): If $A_1 x^{\mathrm{H}} + B_1 y^{\mathrm{L}} \leq b_1$, $c^\top x^{\mathrm{H}} + d_1^\top y^{\mathrm{L}} > \zeta^*$, and $d_1^\top y^{\mathrm{H}} \leq d_1^\top y^{\mathrm{L}}$, then update $(x^* = x^{\mathrm{H}}, y^* = y^{\mathrm{L}}, \zeta^* = c^\top x^{\mathrm{H}} + d_1^\top y^{\mathrm{L}})$ and go to Step 1.

3(d): If $A_1 x^{\mathrm{H}} + B_1 y^{\mathrm{L}} \leq b_1$ and $c^\top x^{\mathrm{H}} + d_1^\top y^{\mathrm{L}} > \zeta^*$, then update $(x^* = x^{\mathrm{H}}, y^* = y^{\mathrm{L}}, \zeta^* = c^\top x^{\mathrm{H}} + d_1^\top y^{\mathrm{L}})$. In either case, go to Step 4.

Step 4 Solve the SP $\mathcal{S}(x^{\mathrm{H}}, y^{\mathrm{H}})$ and let $(\Delta y^{\mathrm{S}}, t^{\mathrm{S}})$ be an optimal solution. Go to Step 5.

Step 5: Let T be the number of positive elements in t^{S} (from either Step 4 or Step 6) and π the set of row indices of the T positive elements in t^{S}. Create T new nodes. For $j = 1, ..., T$, node $N + j$ characterized by $(l^{N+j}, u^{N+j}, z^{N+j})$ is defined as:

$$l_i^{N+j} = \begin{cases} (b_2 - t^{\mathrm{S}})_i + 1, & i = \pi(j); \\ \hat{l}_i, & \text{otherwise.} \end{cases} \tag{4.35}$$

$$u_i^{N+j} = \begin{cases} (b_2 - t^{\mathrm{S}})_i, & i \leq \pi(j) - 1; \\ \hat{u}_i, & \text{otherwise.} \end{cases} \tag{4.36}$$

$$z^{N+j} = \begin{cases} c^\top x^{\mathrm{H}} + d_1^\top y^{\mathrm{H}}, & \text{Step 4 was the previous step;} \\ +\infty, & \text{Step 6 was the previous step.} \end{cases} \tag{4.37}$$

Increase N by T. Go to Step 1.

Step 6: Solve the PBSP $\mathcal{B}(\hat{l}, \hat{u})$.

6(a): If $\mathcal{B}(\hat{l}, \hat{u})$ is infeasible, then stop. The BILP (4.1)-(4.4) is unbounded.

6(b): Let $(x^{\mathrm{S}}, y^{\mathrm{S}}, \Delta y^{\mathrm{S}}, t^{\mathrm{S}})$ be an optimal solution to $\mathcal{B}(\hat{l}, \hat{u})$. If $t^{\mathrm{S}} = 0$, then stop. The BILP (4.1)-(4.4) is infeasible.

6(c): Go to Step 5.

In the following, we establish the correctness and finite termination of $\mathrm{WAlg}^{\mathrm{BILP}}$. First, we define the following parametric MILP, which is denoted as $\mathcal{F}(u)$:

$$\max_{\Delta y, t} \{f(u) = 1^\top t : \text{Constraints (4.22) and (4.23)}; 0 \leq t \leq u; \Delta y \in \mathbb{Z}^{n_2}\}. \tag{4.38}$$

Here the optimal objective value of $\mathcal{F}(u)$ is denoted as $f(u)$. The following lemmas present some important properties of $\mathcal{F}(u)$.

Lemma 13. *For any $u_1 \in \mathbb{R}_+^{m_2}$, $u_2 \in \mathbb{R}_+^{m_2}$, and $u_1 \leq u_2$, we have $f(u_1) \geq f(u_2)$.*

Proof. One only needs to observe that the optimal solution to $\mathcal{F}(u_1)$ must be feasible to $\mathcal{F}(u_2)$, but the reverse statement is not true. □

Lemma 14. *$\{f(u) | u \in \mathbb{Z}_+^{m_2}\}$ is bounded.*

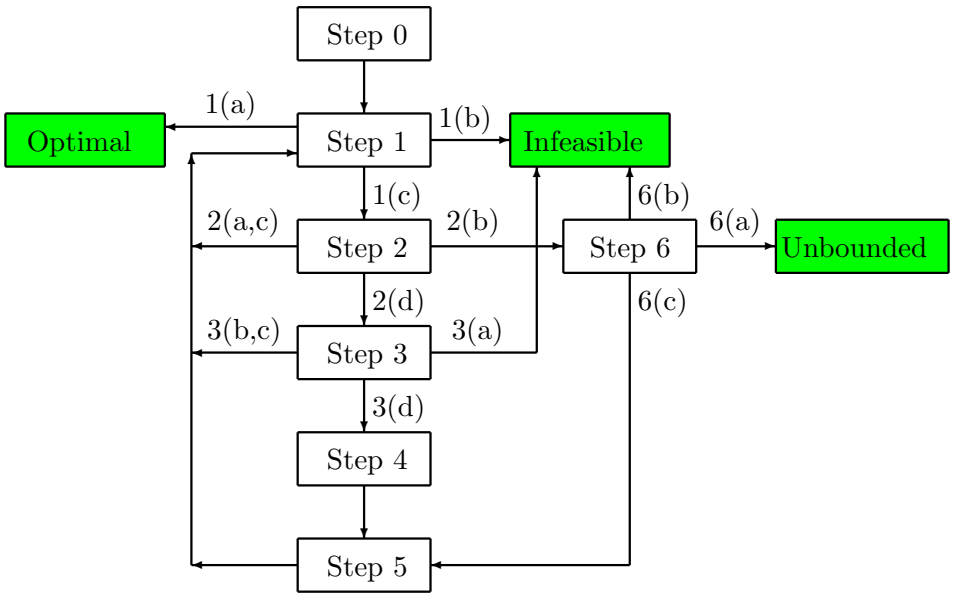

Figure 4.1 Diagram of Alg$^{\text{BILP}}$.

Proof. Prove by contradiction. Suppose $f(u)$ is unbounded, then there must exist an infinite sequence $S_0 = \{u_k \in \mathbb{Z}_+^{m_2}\}, \forall k \in \mathbb{N}$ with $f(u_k) < f(u_l), \forall k < l; k, l \in \mathbb{N}$. Define $G = \bigcup_{i=1}^{m_2} \bigcup_{j=0}^{u_1(i)} G(i,j)$, where $G(i,j) = \{u | u \in \mathbb{Z}_+^{m_2}, u(i) = j\}$. By definition, $S_0 \subseteq G$. By the pigeonhole principle (Jukna and Jukna, 2011), there exist i_1, j_1, and an infinite subsequence $S_1 \subseteq S_0$ such that $S_1 \subseteq G(i_1, j_1)$. Without loss of generality, assume $i_1 = 1$. As such, we get an infinite subset $N_1 \subseteq \mathbb{N}$ such that $S_1 = \{u_k \in \mathbb{Z}_+^{m_2}\}, k \in N_1$ is an infinite sequence with $u_k(1) = j_1, f(u_k) < f(u_l), \forall k < l; k, l \in N_1$. Recursively applying the same principle, we get an infinite subset $N_{m_2} \subseteq N_{m_2-1} \subseteq ... \subseteq N_1$ such that $S_{m_2} = \{u_k \in \mathbb{Z}_+^{m_2}\}, k \in N_{m_2}$ with $u_k(1) = j_1, u_k(2) = j_2, ..., u_k(m_2) = j_{m_2}, f(u_k) < f(u_l), \forall k < l; k, l \in N_{m_2}$. Apparently, S_{m_2} does not exist.

□

Now we present the proof for the correctness and finite termination of WAlg$^{\text{BILP}}$.

Theorem 2. *WAlg^{BILP} finitely terminates with the correct output for* (4.1)-(4.4).

Proof. Our proof consists of five parts: (i) no bilevel feasible solutions is eliminated without

comparison with the incumbent in the algorithm; (ii) Condition 1(a) is sufficient to claim the optimality of a BILP, (iii) Conditions 1(b), 3(a), and 6(b) are sufficient to claim the infeasibility of a BILP, (iv) Condition 6(a) is sufficient to claim the unboundedness of a BILP, and (v) the algorithm will terminate in a finite number of iterations.

Part (i): The branching Step 5 only eliminates the scoop of watermelon flesh, which, by Lemmas 6 and 9, contains no bilevel feasible solutions. All bilevel feasible solutions encountered are compared to the incumbent in Steps 1 and 2 before they either become the new incumbent or are pruned.

Part (ii): Under Condition 1(a), all active nodes have been evaluated, thus the incumbent solution is the optimal one to the BILP.

Part (iii): Under Condition 1(b), all active nodes have been evaluated and no bilevel feasible solution was found, so the BILP must be infeasible. Under Conditions 3(a) and 6(b), the infeasibility of the BILP is ensured by Lemmas 5 and 11, respectively.

Part (iv): By Lemma 10.

Part (v): Lemma 14 implies that the optimal objective value of $\mathcal{S}(x, y)$ is bounded, regardless of the parameters x and y. For that matter, the optimal objective value of $\mathcal{B}(l, u)$ is also bounded, regardless of the parameters l and u. Let K denote a bound for both $\mathcal{S}(x, y)$ and $\mathcal{B}(l, u)$. Then for all $i = 1, ..., m_2$, the possible bounds (l, u) that characterize any node can only lie within $\{-\infty, (b_2)_i - K, (b_2)_i - K + 1, ..., (b_2)_i\}$, which is a finite set. Moreover, the branching Step 5 ensures that no two nodes have the same bounds. Therefore, the branch-and-bound tree of $\text{WAlg}^{\text{BILP}}$ only contains a finite number of possible nodes. □

4.3 Computational Experiments

In this section we report our preliminary computational experience with $\text{WAlg}^{\text{BILP}}$. First, we apply $\text{WAlg}^{\text{BILP}}$ on the classic example from Moore and Bard (1990). In Step 2, the HPP solution $(2, 4)$ is obtained, which is found to be a seed in Step 3. In Step 4, a scoop of watermelon flesh is computed, which is colored as red in Figure 3.2. In fact, this scoop is able to encloses all watermelon seeds (solid squares) in this particular example, and all other integer solutions (empty squares or heart) are bilevel feasible. Apparently, the optimal solution (2,2) is easily

found in the next iteration. In contrast, the algorithm in Moore and Bard (1990) takes a lot more iterations to find the same optimal solution.

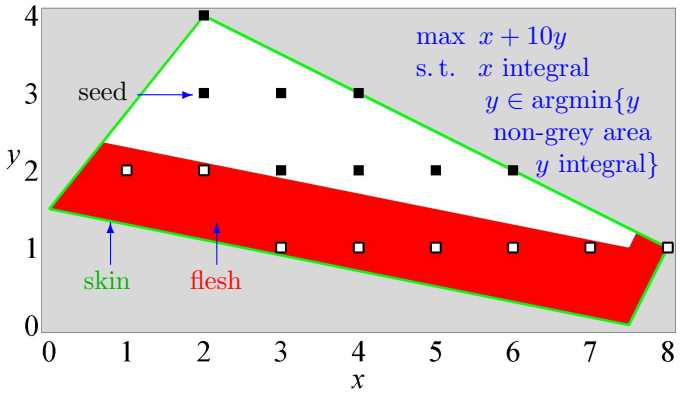

Figure 4.2 A BILP example from Moore and Bard (1990)

We also test the algorithm using eleven randomly generated instances. For each case, we first set up the sizes m_1, m_2, n_1, n_2 and boundaries that values in each parameter range from and then randomly output one. Table 4.1 summarizes some details about these instances including dimension, boundedness of HPP, and optimal objective value. The algorithm is implemented in Matlab using TOMLAB/CPLEX as the MILP solver. Computational experiments are executed on a desktop computer with a 3 GB ram and a 2.4 GHz CPU. Computational results are summarized in Table 4.2. Three searching strategies are used in Step 1(c) to select the new node to visit: depth-first, breadth-first, and largest-z-first. Their computational time (in the hour-minute-second format) and number of nodes visited are reported in Table 4.2. Although no strategy is the clear dominator, the largest-z-first strategy appears to be more promising than the other two.

4.4 Conclusions and Discussions

This paper presents the watermelon algorithm for BILP, which makes several contribution to the literature. First, it is based on a novel approach. Rather than using branch-and-bound or cutting plane methods to approximate the non-convex bilevel feasibility region from the outside, as most previous algorithms do, it removes convex bilevel infeasibility regions from

Table 4.1 Eleven test instances.

Instance	m_1	n_1	m_2	n_2	HPP	ζ^*
1	15	5	3,010	10	Bounded	$-\infty$
2	15	5	205	5	Bounded	546
3	55	5	105	5	Bounded	803
4	15	5	305	5	Bounded	640
5	10	5	2,005	5	Unbounded	310
6	10	5	505	5	Unbounded	∞
7	100	50	100	50	Bounded	75
8	105	55	100	50	Bounded	86
9	110	60	110	60	Bounded	86
10	120	70	110	60	Bounded	83
11	150	100	150	100	Bounded	189

the interior. Second, as far as we are aware of, it is the first algorithm that explicitly promises to solve all possible BILPs, be it infeasible, unbounded, or optimal. Third, we draw analogies between BILP and watermelon, which inspired the discovery of several interesting properties of BILP in metaphorically described yet mathematically proved lemmas. Fourth, we provide a proof of the correctness and finite termination of the algorithm.

A watermelon algorithm for the bilevel mixed integer linear program (BMILP) appears to be a natural extension. However, the possibility of the unattainable supremum remains a challenge. One alternative is to seek the ϵ-optimal solution (Köppe et al., 2009), which promises to be within an ϵ-neighborhood of the supremum. Another extension is to apply a similar idea to the watermelon algorithm on BLP, the continuous relaxation of BILP. There are many previous BLP algorithms (Audet et al., 2007a, 1997, 2007b; Bard and Moore, 1990; Brotcorne et al., 2009; Hansen et al., 1992; Júdice and Faustino, 1992; Önal, 1993; Wang et al., 2007a), but most are based on using the complementarity slackness of the lower level linear program as the upper level constraints. The watermelon algorithm, on the contrary, will create a scoop that carves the entire interior out and leaves the skins under examination.

Table 4.2 Computational results.

Instance	CPU time			# Nodes visited		
	depth	breadth	best-z	depth	breadth	largest-z
1	24m-26s	34m-10s	out-of-memory	11,627	11,627	out-of-memory
2	3m-44s	59s	1m	10,062	2,580	2,580
3	7m-35s	4s	3s	26,425	194	160
4	18m-16s	22m-29s	16m-11s	31,863	24,770	16,822
5	2h-32m-46s	7m-19s	8m-41s	31,221	2,719	2,719
6	11m-23s	30m-31s	42s	15,696	10,928	870
7	3h-7m-48s	1h-59m-56s	36m-47s	10,642	8,194	4,074
8	34m-4s	18m-42s	25m-51s	953	514	758
9	1h-12m-14s	2m-27s	2m-40s	1,464	122	160
10	1h-2m-48s	11m-3s	4m-55s	1,755	234	176
11	17s	17s	17s	2	2	2

CHAPTER 5. CONCLUSION

The research presented in the three papers here makes clear contributions to the area of bilevel optimization algorithms and applications. We will discuss these contributions in detail as follows.

In Chapter 2, we present an optimization approach to stacking desirable genes scattered among multiple inbred lines into one population of target genotypes. The issue of biological diversity is also taken into account by preserving desirable alleles of all allelic variations. We formulate this problem as a multi-objective integer programming program, which yields Pareto optimal stacking strategies with respect to two criteria: the likelihood of successfully obtaining the target population of genotypes and the number of generations associated with a stacking strategy. Numerical examples demonstrate the feasibility of using the proposed approach to obtain Pareto optimal stacking strategies for smaller instances within one minute.

Our approach makes a distinct contribution to the animal and plant breeding practices and theoretical studies on the genetic architecture of complex traits. To our best knowledge, this is the first study that formulates the gene stacking problem as a multi-objective optimization model. This approach takes advantage of the state-of-the-art algorithms and computer solvers that can efficiently search the entire exponentially exploding (Servin et al., 2004) solution space and provide solutions that can be mathematically proved to be optimal. The flexibility to address biological diversity in the model also extends the model's applicability to a wider range of situations, in which all alleles cannot be simply differentiated as desirable or undesirable but also exhibit distinctions in their desirability.

In Chapter 3, we propose an exact algorithm for the BMILP under three simplifying assumptions. Our study makes the following contributions to the existing literature. First, $\text{Alg}^{\text{BMILP}}$ relies on weaker simplifying assumptions than existing algorithms. Second, our algorithm ex-

plicitly considers all possible outcomes of a BMILP, including infeasible, unbounded, and finite optimal. Third, we provide a proof of the correctness and finite termination of $\text{Alg}^{\text{BMILP}}$. Fourth, we demonstrate the efficiency of our algorithm by reporting our computational results on small- to medium-sized instances. Last but not least, we have created a library of BMILP test instances for benchmark and comparison of future algorithms.

In Chapter 4, we present the watermelon algorithm for BILP, which makes several contributions to the literature. First, it is based on a novel approach. Rather than using branch-and-bound or cutting plane methods to approximate the non-convex bilevel feasibility region from the outside, as most previous algorithms do, it removes convex bilevel infeasible regions from the interior. Second, as far as we are aware of, it is the first algorithm that explicitly promises to solve all possible BILPs, be it infeasible, unbounded, or optimal. Third, we draw analogies between BILP and watermelon, which inspired the discovery of several interesting properties of BILP in metaphorically described yet mathematically proved lemmas. Fourth, we provide a proof of the correctness and finite termination of the algorithm.

Our future work lies in the following directions:

- As for the first paper, to reformulate the plant breeding problem under a bilevel optimization schema could be more practical and closer to the nature. We can try to interpret the plant breeding problem in a hierarchical frame as follows: in the upper level breeders select plants and provide breeding environments for certain purposes, while in the lower level the breeding outcome is autonomously governed by the rules of genetics and probability. This hierarchical frame could be better captured by a nested maximum likelihood function rather than being traded with the objective of the breeder on a Pareto frontier. In fact, bilevel optimization model has been applied to a related genetic engineering problem (Burgard et al., 2003).

- As for the second paper, natural extension should be to relax the first two assumptions we made. BMILP would become much more complicated if we allow both levels have a mixed integer setting. In this case, $\text{Alg}^{\text{BMILP}}$ should be modified to output an ϵ-optimal solution (Köppe et al., 2009) when the supremum is not attainable. To relax the second

assumption, we may need to add some essential changes to $\text{Alg}^{\text{BMILP}}$. Just as shown in the Example 4, $\text{Alg}^{\text{BMILP}}$ would fail to terminate finitely when feasible region for the upper level variables is unbounded. The third assumption can be easily extended to $A_2 \in \mathbb{Q}^{m_2 \times n_1}$, which was made explicitly or inexplicitly in most current related literature.

- As for the third paper, it would be interesting to extend $\text{WAlg}^{\text{BILP}}$ to the case of BLP where variables in both levels are all continuous. Most of existing algorithms solving BLP (Audet et al., 2007a, 1997, 2007b; Bard and Moore, 1990; Brotcorne et al., 2009; Hansen et al., 1992; Júdice and Faustino, 1992; Önal, 1993; Wang et al., 2007a) are based on the framework that is to convert the lower level optimality constraint to linear complimentary constraints using KKT condition. By applying $\text{WAlg}^{\text{BILP}}$, we can potentially get rid of the classic framework completely and invent a novel approach to solving BLP. Note that optimistic assumption is inexplicitly made in the formulation (4.1)-(4.4). It might deserve our efforts to consider the case when pessimistic assumption is made. That would be a good start for us to extend $\text{WAlg}^{\text{BILP}}$ to solve the complicated trilevel optimization problems.

BIBLIOGRAPHY

Achterberg, T., Koch, T., and Martin, A. (2006). MIPLIB 2003. *Operations Research Letters*, 34:361–372.

Aiyoshi, E. and Shimizu, E. (1981). Hierarchical decentralized systems and its new solution by a barrier method. *IEEE Transactions on Systems, Man, and Cybernetics*, 11:444–449.

Aiyoshi, E. and Shimizu, E. (1984). A solution method for the static constrained Stackelberg problem via penalty method. *IEEE Transactions on Automatic Control*, 29:1111–1114.

Anderson, J., Chao, S., and Liu, S. (2008). Molecular breeding using a major QTL for Fusarium head blight resistance in wheat. *Crop Science*, 48.

Audet, C., Haddad, J., and Savard, G. (2007a). Disjunctive cuts for continuous linear bilevel programming. *Optimization Letters*, 1:259–267.

Audet, C., Hansen, P., Jaumard, B., and Savard, G. (1997). Links between linear bilevel and mixed 0-1 programming problems. *Journal of Optimization Theory and Applications*, 93(2):273–300.

Audet, C., Savard, G., and Zghal, W. (2007b). New branch-and-cut algorithm for bilevel linear programming. *Journal of Optimization Theory and Applications*, 134:353–370.

Bard, J. (1983). Coordination of a multidivisional organization through two levels of management. *Omega*, 11(5):457–468.

Bard, J. (1988). Convex two-level optimization. *Mathematical Programming*, 40:15–27.

Bard, J. and Falk, J. (1982). An explicit solution to the multi-level programming problem. *Computers and Operations Research*, 9:77–100.

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

Bard, J. and Moore, J. (1992). An algorithm for the discrete bilevel programming problem. *Naval Research Logistics*, 39:419–435.

Bard, J., Plummer, J., and Claude Sourie, J. (2000). A bilevel programming approach to determining tax credits for biofuel production1. *European Journal of Operational Research*, 120(1):30–46.

Bellman, R. (1957). *Dynamic Programming*. Princeton University Press, Princeton, New Jersey.

Ben-Ayed, O. (1993). Bilevel linear programming. *Computers & Operations Research*, 20(5):485–501.

Ben-Ayed, O. and Blair, C. (1990). Computational difficulties of bilevel linear programming. *Operations Research*, 38(3):556–560.

Bernardo, R. and Charcosset, A. (2006). Usefulness of gene information in marker-assisted recurrent selection: A simulation appraisal. *Crop Science*, 46:614–621.

Bernardo, R. and Yu, J. (2007). Prospects for genomewide selection for quantitative traits in maize. *Crop Science*, 47:1082–1090.

Bialas, W. and Karwan, M. (1984). Two-level linear programming. *Management Science*, 30(8):1004–1020.

Birge, J. and Louveaux, F. (1997). *Introduction to stochastic programming*. Springer, New York.

Blickle, T., Teich, J., and Thiele, L. (1998). System-level synthesis using evolutionary algorithms. *Design Automation for Embedded Systems*, 3:23–58.

Bonnet, D., Rebetzke, G., and Spielmeyer, W. (2005). Strategies for efficient implementation of molecular markers in wheat breeding. *Molecular Breeding*, 15:75–85.

Bouri, A., Martel, J. M., and Chabchoub, H. (2002). A multi-criterion approach for selecting attractive portfolio. *Journal of Multi-Criteria Decision Analysis*, 11(4-5):269–277.

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

Bracken, J. and McGill, J. (1974). Defense applications of mathematical programs with optimization problems in the constraints. *Operations Research*, 22:1086–1096.

Bracken, J. and McGill, J. (1978). Production and marketing decisions with multiple objectives in a competitive environment. *Journal of Optimization Theory and Applications*, 24:449–458.

Brotcorne, L., Hanafi, S., and Mansi, R. (2009). A dynamic programming algorithm for the bilevel knapsack problem. *Operations Research Letters*, 37(3):215–218.

Brotcorne, L., Labb, M., Marcotte, P., and Savard, G. (2001). A bilevel model for toll optimization on a multicommodity transportation network. *Transportation Science*, 35:1–14.

Burgard, A., Pharkya, P., and Maranas, C. (2003). Optknock: A bilevel programming framework for identifying gene knockout strategies for microbial strain optimization. *Biotechnology and Bioengineering*, 84:647–657.

Cahill, D. and Schmidt, D. (2004). Use of marker assisted selection in a product development breeding program. *Proceedings of the 4th International Crop Science Congress*.

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

Colson, B., Marcotte, P., and Savard, G. (2007). An overview of bilevel optimization. *Annals of Operations Research*, 153:235–256.

Concibido, V., Denny, R., Boutin, S., Hautea, R., Orf, J., and Young, N. (1994). Dna marker analysis of loci underlying resistance to soybean cyst nematode (heterodera glycines ichinohe). *Crop Science*, 34:240–246.

Constantin, I. and Florian, M. (1995). Optimizing frequencies in a transit network: A nonlinear bi-level programming approach. *International Transactions in Operational Research*, 2(2):149–164.

Cote, J., Marcotte, P., and Savard, G. (2003). A bilevel modelling approach to pricing and fare optimisation in the airline industry. *Journal of Revenue and Pricing Management*, 2(1):23–36.

Darwin, C. (1872). *The origin of species*. John Murray, London.

Dekkers, J. (2007). Prediction of response to marker-assisted and genomic selection using selection index theory. *Journal of Animal Breeding and Genetics*, 124:331–341.

Dempe, S. (2001). Discrete bilevel optimization problems. Technical report, Institut fur Wirtschaftsinformatik, Universitat Leipzig, Leipzig, Germany.

Dempe, S., Kalashnikov, V., and Ríos-Mercado, R. (2005). Discrete bilevel programming: Application to a natural gas cash-out problem. *European Journal of Operational Research*, 166(2):469–488.

DeNegre, S. and Ralphs, T. (2009). A branch-and-cut algorithm for integer bilevel linear programs. *Operations Research and Cyber-Infrastructure*, 47:65–78.

Deolalikar, V. (2010). P $\neq$ NP. Technical report, HP Research Labs.

Eathington, S., Crosbie, T., Edwards, M., Reiter, R., and Bull, J. (2007). Molecular markers in a commercial breeding program. *Crop Science*, 47:154–163.

Edmunds, T. and Bard, J. (1991). Algorithms for nonlinear bilevel mathematical programs. *IEEE Transactions on Systems, Man and Cybernetics*, 21(1):83–89.

Edmunds, T. and Bard, J. (1992). An algorithm for the mixed-integer nonlinear bilevel programming problem. *Annals of Operations Research*, 34:149–162.

Eisenbrand, F. and Shmonin, G. (2008). Parametric integer programming in fixed dimension. *Mathematics of Operations Research*, 33(4):839–850.

Fampa, M., Barroso, L., Candal, D., and Simonetti, L. (2008). Bilevel optimization applied to strategic pricing in competitive electricity markets. *Computational Optimization and Applications*, 39(2):121–142.

Fortuny-Amat, J. and McCarl, B. (1981). A representation and economic interpretation of a two-level programming problem. *Journal of the Operational Research Society*, pages 783–792.

Gao, Z., Wu, J., and Sun, H. (2005). Solution algorithm for the bi-level discrete network design problem. *Transportation Research Part B: Methodological*, 39(6):479–495.

Gümüş, Z. and Floudas, C. (2005). Global optimization of mixed-integer bilevel programming problems. *Computational Management Science*, 2(3):181–212.

Hansen, P., Jaumard, B., and Savard, G. (1992). New branch-and-bound rules for linear bilevel programming. *SIAM Journal on Scientific and Statistical Computing*, 13:1194–1217.

He, Y., Wang, L., and Wang, J. (2011). Cap-and-trade vs. carbon taxes: A quantitative comparison from a generation expansion planning perspective. *Computers & Industrial Engineering*, (0):–.

Hillier, F. and Lieberman, G. (2005). *Introduction to Operations Research*. McGraw-Hill, New York.

Howes, N., Woods, S., and Townley-Smith, T. (1998). Simulations and practical problems of applying multiple marker assisted selection and doubled haploids to wheat breeding programs. *Euphytica*, 100:225–230.

Júdice, J. and Faustino, A. (1992). A sequential lcp method for bilevel linear programming. *Annals of Operations Research*, 34(1):89–106.

Jukna, S. and Jukna, S. (2011). The pigeonhole principle. In *Extremal Combinatorics*, Texts in Theoretical Computer Science. An EATCS Series, pages 53–75. Springer Berlin Heidelberg.

Jung, J., Fan, R., and Jin, L. (2005). Combined linkage and association mapping of quantitative trait loci by multiple markers. *Genetics*, 170:881–898.

Kall, P. and Wallace, S. (1994). *Stochastic Programming*. Wiley-Interscience, New York.

Kebede, H., Subudhi, P., Rosenow, D., and Nguyen, H. (2001). Quantitative trait loci influencing drought tolerance in grain sorghum (Sorghum bicolor L. Moench). *Theoretical and Applied Genetics*, 103:266–276.

Köppe, M., Queyranne, M., and Ryan, C. T. (2009). A parametric integer programming algorithm for bilevel mixed integer programs. *Journal of Optimization Theory and Applications*, 146:11.

Labbe, M., Marcotte, P., and Savard, G. (1998). A bilevel model of taxation and its application to optimal highway pricing. *Management Science*, 44:1608–1622.

Lande, R. and Thompson, R. (1990). Efficiency of marker-assisted selection in the improvement of quantitative traits. *Genetics*, 124:743–756.

Lange, C. and Whittaker, J. (2001). On prediction of genetic values in marker-assisted selection. *Genetics*, 159:1375–1381.

Lim, C. and Smith, J. (2007). Algorithms for discrete and continuous multicommodity flow network interdiction problems. *IIE Transactions*, 39(1):15–26.

Mackay, T. F. C. (2004). The genetic architecture of quantitative traits: lessons from Drosophila. *Current Opinion in Genetics & Development*, 14:253–257.

Moore, J. and Bard, J. (1990). The mixed integer linear bilevel programming problem. *Operations Research*, 38:911–921.

Morton, D., Pan, F., and Saeger, K. (2007). Models for nuclear smuggling interdiction. *IIE Transactions*, 39(1):3–14.

Murty, K. (1995). *Operations Research: Deterministic Optimization Models*. Prentice-Hall, Englewood Cliffs, NJ.

Önal, H. (1993). A modified simplex approach for solving bilevellinear programming problems. *European Journal of Operational Research*, 67(1):126–135.

Papadimitriou, C. and Yannakakis, M. (2000). On the approximability of trade-offs and optimal access of web sources. *Foundations of Computer Science, IEEE Annual Symposium on*, 0:86.

Podlich, D., Winkler, C., and Cooper, M. (2004). Mapping as you go: An effective approach to marker-assisted-selection of complex traits. *Crop Science*, 44:1560–1571.

Pumphrey, M., Bernardo, R., and Anderson, J. (2007). Validating the Fhb1 QTL for Fusarium head blight resistance in near-isogenic wheat lines developed from breeding populations. *Crop Science*, 47:200–206.

Puterman, M. (1994). *Markov Decision Processes*. John Wiley & Sons, Inc., New York.

Rajesh, J., Gupta, S., Rangaiah, G., and Ray, A. (2001). Multi-objective optimization of industrial hydrogen plants. *Chemical Engineering Science*, 56(3):999 – 1010.

Salmeron, J., Wood, K., and Baldick, R. (2004). Analysis of electric grid security under terrorist threat. *Power Systems, IEEE Transactions on*, 19(2):905–912.

Sarma, K. and Adeli, H. (2001). Bilevel parallel genetic algorithms for optimization of large steel structures. *Computer-Aided Civil and Infrastructure Engineering*, 16(5):295–304.

Scaparra, M. and Church, R. (2008). A bilevel mixed-integer program for critical infrastructure protection planning. *Computers & Operations Research*, 35(6):1905–1923.

Servin, B., Martin, O., Mézard, M., and Hospital, F. (2004). Toward a theory of marker-assisted gene pyramiding. *Genetics*, 168:513–523.

Tanksley, S. and Hewitt, J. (1988). Use of molecular markers in breeding for soluble solids content in tomatoa re-examination. *Theoretical and Applied Genetics*, 75:811–823.

Tseng, C. H. and Lu, T. W. (1990). Minimax multiobjective optimization in structural design. *International Journal for Numerical Methods in Engineering*, 30(6):1213–1228.

Vicente, L. and Calamai, P. (1994). Bilevel and multilevel programming: A bibliography review. *Journal of Global Optimization*, 5(3):291–306.

Vicente, L., Savard, G., and Júdice, J. (1996). Discrete linear bilevel programming problem. *Journal of Optimization Theory and Applications*, 89:597–614.

Wang, G., Wang, X., Wan, Z., and Jia, S. (2007a). An adaptive genetic algorithm for solving bilevel linear programming problem. *Applied Mathematics and Mechanics*, 28(12):1605–1612.

Wang, J., Chapman, S., Bonnett, D., Rebetzke, G., and Crouch, J. (2007b). Application of population genetic theory and simulation models to efficiently pyramid multiple genes via marker-assisted selection. *Crop Science*, 47:582–588.

Wang, L. (2009). Cutting plane algorithms for the inverse mixed integer linear programming problem. *Operations Research Letters*, 37(2):114–117.

Wen, U.-P. and Hsu, S.-T. (1991). Linear bi-level programming problems – a review. *The Journal of the Operational Research Society*, 42(2):125–133.

Whittaker, J. (2001). Marker assisted selection and introgression for mapping quantitative trait loci in line crosses using flanking markers. *Heredity*, 69:315–324.

Wu, R., Ma, C., and Casella, G. (2002). Joint linkage and linkage disequilibrium mapping of quantitative trait loci in natural populations. *Genetics*, 160:779–792.

Yao, Y., Edmunds, T., Papageorgiou, D., and Alvarez, R. (2007). Trilevel optimization in power network defense. *IEEE Transaction on Systems, Man, and Cybernetics*, 37(4):712–718.

Zhou, Y., Wang, L., and McCalley, J. D. (2011). Designing effective and efficient incentive policies for renewable energy in generation expansion planning. *Applied Energy*, 88(6):2201 – 2209.