


5-2014

# Improved Mixed-Integer Models of a Two-Dimensional Cutting Stock Problem

William Lassiter

Clemson University, [wlassit@g.clemson.edu](mailto:wlassit@g.clemson.edu)

Follow this and additional works at: [https://tigerprints.clemson.edu/all\\_theses](https://tigerprints.clemson.edu/all_theses)

 Part of the [Applied Mathematics Commons](#), and the [Operations Research, Systems Engineering and Industrial Engineering Commons](#)

---

## Recommended Citation

Lassiter, William, "Improved Mixed-Integer Models of a Two-Dimensional Cutting Stock Problem" (2014). *All Theses*. 1944.  
[https://tigerprints.clemson.edu/all\\_theses/1944](https://tigerprints.clemson.edu/all_theses/1944)

This Thesis is brought to you for free and open access by the Theses at TigerPrints. It has been accepted for inclusion in All Theses by an authorized administrator of TigerPrints. For more information, please contact [kokeefe@clemson.edu](mailto:kokeefe@clemson.edu).

IMPROVED MIXED-INTEGER MODELS OF A TWO-DIMENSIONAL  
CUTTING STOCK PROBLEM

---

A Thesis  
Presented to  
the Graduate School of  
Clemson University

---

In Partial Fulfillment  
of the Requirements for the Degree  
Master of Science  
Mathematical Sciences

---

by  
William B. Lassiter  
May 2014

---

Accepted by:  
Dr. Warren Adams, Committee Chair  
Dr. Matthew Saltzman  
Dr. Akshay Gupte

# Abstract

This paper is concerned with a family of two-dimensional cutting stock problems that seeks to cut rectangular regions from a finite collection of sheets in such a manner that the minimum number of sheets is used. A fixed number of rectangles are to be cut, with each rectangle having a known length and width. All sheets are rectangular, and have the same dimension. We review two known mixed-integer mathematical formulations, and then provide new representations that both economize on the number of discrete variables and tighten the continuous relaxations. A key consideration that arises repeatedly in all models is the enforcement of disjunctions that a vector must lie in the union of a finite collection of polytopes. Computational results demonstrate a relative performance of the different formulations.

# Table of Contents

Title Page . . . . . i

Abstract . . . . . ii

List of Tables . . . . . iv

1 Introduction . . . . . 1

2 Modeling Disjunctions . . . . . 8

3 Proposed Models . . . . . 15

4 Computational Experience . . . . . 20

5 Conclusions and Discussion . . . . . 26

Appendix: Cutting Rectangle and Sheet Dimensions . . . . . 28

Bibliography . . . . . 29

# List of Tables

4.1	SP1 ( $n = 7$ rectangles, $m = 4$ sheets) . . . . .	23
4.2	SP2 ( $n = 18$ rectangles, $m = 4$ sheets) . . . . .	24
4.3	SP1' ( $n = 7$ rectangles, $m = 1$ sheet) . . . . .	25
4.4	SP2' ( $n = 18$ rectangles, $m = 1$ sheet) . . . . .	25

# Chapter 1

## Introduction

Given a collection of  $m$  rectangular sheets of length  $L$  and width  $W$ , the two-dimensional cutting stock problem of concern is to cut, from amongst these sheets, a collection of  $n$  rectangles so that the minimum number of sheets is used. Here, each rectangle  $R_i$  for  $i = 1, \dots, n$  has known length  $\ell_i$  and width  $w_i$ . Each rectangle is to be cut so that either the length or the width is parallel to the  $x$ -axis.

This problem, and variants thereof, have received attention by different authors [3, 4, 5]. The particular problem of concern was studied in [5], within which is found two different mixed-integer linear formulations. The main contribution of this paper is to encompass these formulations within a general disjunctive framework that allows for the devising of different forms, and to perform computational tests to assess the relative merits.

There are two key modeling challenges relative to this problem. The first, which is shared by related problems in [3, 4], is to ensure that no overlap exists between cut rectangles; that is, that no region found on any sheet is used by more than one rectangle. The second challenge is to ensure that each rectangle is cut from exactly one sheet.

To motivate the construction of a model, envision that the sheets are arranged vertically in a 2-dimensional plane so that the bottom-most sheet has its lower left corner situated at the origin, and so that the width of each sheet is parallel to the  $x$ -axis. The sheets are stacked so that, given any two which are adjacent, the top of the lower sheet is aligned flush with the bottom of the upper

sheet. Now, for each rectangle  $R_i$ , define an ordered pair of continuous decision variables  $(x_i, y_i)$  to denote the coordinates of the lower left corner of the region from which  $R_i$  is to be cut. Also define a binary decision variable  $s_i$  to denote the orientation of rectangle  $R_i$ , so that  $s_i = 1$  if the side having length  $\ell_i$  is parallel to the  $x$ -axis, and  $s_i = 0$  otherwise. Then the right side of rectangle  $R_i$  will be at value  $x_i + w_i + s_i(\ell_i - w_i)$  and the top of rectangle  $R_i$  will be at value  $y_i + \ell_i + s_i(w_i - \ell_i)$ . Of course, if  $\ell_i = w_i$  for any rectangle  $R_i$ , then the variable  $s_i$  is not needed.

The first challenge of preventing rectangle overlap is addressed using a method found in [3]. For every pair of rectangles  $R_i$  and  $R_j$  with  $i < j$ , the following disjunctive statement prevents overlap:

$$\begin{aligned} & (R_i \text{ is to the right of } R_j) \text{ or } (R_i \text{ is above } R_j) \text{ or} \\ & (R_i \text{ is to the left of } R_j) \text{ or } (R_i \text{ is below } R_j). \end{aligned}$$

This statement can be rewritten in terms of  $(x_i, x_j, y_i, y_j, s_i, s_j)$  as follows

$$\begin{aligned} x_i & \geq x_j + w_j + s_j(\ell_j - w_j) \text{ or } y_i \geq y_j + \ell_j + s_j(w_j - \ell_j) \text{ or} \\ x_i & \leq x_j - w_i - s_i(\ell_i - w_i) \text{ or } y_i \leq y_j - \ell_i - s_i(w_i - \ell_i). \end{aligned} \tag{1.1}$$

To enforce (1.1), define four binary variables, say  $a_{ij}, b_{ij}, c_{ij}, d_{ij}$ , to associate with the four propositions of (1.1) in a one-to-one fashion so that the  $k^{\text{th}}$  proposition holds true when the  $k^{\text{th}}$  variable equals 1. Then restrict the sum of the four binary variables to equal 1 to ensure that at least one

proposition holds true. Specifically, for each  $(i, j), i < j$ , define the set  $S_{ij}$ .

$$S_{ij} = \{(x_i, x_j, y_i, y_j, s_i, s_j, a_{ij}, b_{ij}, c_{ij}, d_{ij}) :$$

$$x_i \geq x_j + w_j + s_j(\ell_j - w_j) - W(1 - a_{ij}), \quad (1.2)$$

$$y_i \geq y_j + \ell_j + s_j(w_j - \ell_j) - mL(1 - b_{ij}), \quad (1.3)$$

$$x_i \leq x_j - w_i - s_i(\ell_i - w_i) + W(1 - c_{ij}), \quad (1.4)$$

$$y_i \leq y_j - \ell_i - s_i(w_i - \ell_i) + mL(1 - d_{ij}), \quad (1.5)$$

$$a_{ij} + b_{ij} + c_{ij} + d_{ij} = 1, \quad (1.6)$$

$$a_{ij}, b_{ij}, c_{ij}, d_{ij} \text{ binary}\}$$

Each set  $S_{ij}$  operates as follows. The inequalities (1.2), (1.3), (1.4), and (1.5) correspond, in order, to the four propositions of (1.1). That variable from amongst  $\{a_{ij}, b_{ij}, c_{ij}, d_{ij}\}$  which is selected to realize value 1 activates the associated inequality, and thereby enforces the associated proposition. The scalars  $W$  and  $mL$  are sufficiently large that the remaining three inequalities within (1.2)–(1.5) are redundant.

The second challenge of ensuring that each rectangle is cut from exactly one sheet is satisfied by [5] in two different ways, in two different formulations. The first formulation is Problem P1 below. For convenience, we adopt the notation that  $\mathcal{N} \equiv \{(i, j) : i \in N, j \in N, i < j\}$  denotes the set of  $n(n - 1)/2$  distinct pairs of rectangles  $R_i$  and  $R_j$  that are to be cut. We also let the index  $i$  run from 1 to  $n$ , and the index  $k$  run from 1 to  $m$ , unless stated otherwise.



P1: minimize  $Y$

$$\text{subject to } (x_i, x_j, y_i, y_j, s_i, s_j, a_{ij}, b_{ij}, c_{ij}, d_{ij}) \in S_{ij} \forall (i, j) \in \mathcal{N} \quad (1.7)$$

$$y_i \leq L \left( \sum_{k=1}^m Q_{ik} k \right) - \ell_i - s_i(w_i - \ell_i) \forall i \quad (1.8)$$

$$y_i \geq L \left( \sum_{k=1}^m Q_{ik} (k-1) \right) \forall i \quad (1.9)$$

$$1 = \sum_{k=1}^m Q_{ik} \forall i \quad (1.10)$$

$$W \geq x_i + w_i + s_i(\ell_i - w_i) \forall i \quad (1.11)$$

$$Y \geq y_i + \ell_i + s_i(w_i - \ell_i) \forall i \quad (1.12)$$

$$x_i \geq 0, y_i \geq 0 \forall i, Q_{ik} \text{ binary } \forall (i, k) \quad (1.13)$$

Problem P1 operates as follows. Restrictions (1.7) prevent rectangles from overlapping, as explained above. For each  $i \in N$ , the two inequalities of (1.8) and (1.9), together with the equation of (1.10) and binary restrictions on  $Q_{ik}$  for all  $k$  of (1.13), ensure that rectangle  $R_i$  is cut from exactly one sheet. These restrictions enforce that, for some sheet  $k$ , the lower height  $y_i$  of  $R_i$  is bounded below by the lower height  $L(k-1)$  of sheet  $k$  and the upper height  $y_i + \ell_i + s_i(w_i - \ell_i)$  of  $R_i$  is bounded above by the upper height  $L_k$  of sheet  $k$ . Inequalities (1.11) enforce the width restriction on each sheet, and inequalities (1.12) record the minimum overall height  $Y$ . (The paper [5] does not include the equations (1.10) nor the nonnegativity restrictions on the variables  $x_i$ , but these omissions appear to be oversights and do not affect the merits of the paper.) Problem P1 has  $5n(n+1)/2$  constraints in (1.7)–(1.12),  $2n$  nonnegative continuous variables  $(x_i, y_i)$  and a continuous variable  $Y$ , and  $n(2n+m-1)$  binary variables.

The second formulation of [5] uses a reduced number of binary variables. This reduction is accomplished in two steps. The first step applies a method of [4] to replace the four binary variables  $a_{ij}, b_{ij}, c_{ij}$ , and  $d_{ij}$  of each set  $S_{ij}$  with two binary variables  $\alpha_{ij}$  and  $\beta_{ij}$ . The replacement

is accomplished by defining new sets  $S'_{ij}$  as follows.

$$\begin{aligned}
S'_{ij} = \{ & (x_i, x_j, y_i, y_j, s_i, s_j, \alpha_{ij}, \beta_{ij}) : \\
& x_i \geq x_j + w_j + s_j(\ell_j - w_j) - W(\alpha_{ij} + \beta_{ij}), \\
& y_i \geq y_j + \ell_j + s_j(w_j - \ell_j) - mL((1 - \alpha_{ij}) + \beta_{ij}), \\
& x_i \leq x_j - w_i - s_i(\ell_i - w_i) + W(\alpha_{ij} + (1 - \beta_{ij})), \\
& y_i \leq y_j - \ell_i - s_i(w_i - \ell_i) + mL((1 - \alpha_{ij}) + (1 - \beta_{ij})), \\
& \alpha_{ij}, \beta_{ij} \text{ binary} \}
\end{aligned}$$

Here, for each of the four possible binary realizations of the variables  $\alpha_{ij}$  and  $\beta_{ij}$ , exactly one of the expressions  $(\alpha_{ij} + \beta_{ij})$ ,  $((1 - \alpha_{ij}) + \beta_{ij})$ ,  $(\alpha_{ij} + (1 - \beta_{ij}))$ , and  $((1 - \alpha_{ij}) + (1 - \beta_{ij}))$  will be equal to 0, and the other three expressions will be greater than or equal to 1. That expression equalling 0 has the associated inequality active. This activation of an inequality allows restrictions (1.7) of Problem P1 to be replaced with

$$(x_i, x_j, y_i, y_j, s_i, s_j, \alpha_{ij}, \beta_{ij}) \in S'_{ij} \quad \forall (i, j) \in \mathcal{N}, \quad (1.14)$$

reducing the size of P1 by  $n(n - 1)$  binary variables and  $n(n - 1)/2$  constraints.

The second step reformulates (1.8) and (1.9) so as to replace the  $nm$  binary variables  $Q_{ik}$  with  $n \lceil \log_2 m \rceil$  binary variables. To explain, we adopt the notation of [1], because a method of [1] will be shown in the following section to improve upon that of [5]. For each sheet  $k$ , define the binary vector  $\mathbf{v}_k \in \mathbb{R}^{\lceil \log_2 m \rceil}$  in terms of the base-2 expansion of the integer  $k - 1$  so that

$$k - 1 = \sum_{j=1}^{\lceil \log_2 m \rceil} v_{kj} 2^{j-1}, \quad (1.15)$$

where, for each sheet  $k$ , we let  $v_{kj}$  denote entry  $j$  of  $\mathbf{v}_k$ . Then define  $n$  linear functions  $A_k(\mathbf{u})$  of

variables  $\mathbf{u} \in \mathbb{R}^{\lceil \log_2 m \rceil}$  in terms of the  $m$  binary vectors  $\mathbf{v}_k$  so that

$$A_k(\mathbf{u}) = \sum_{j:\mathbf{v}_{kj}=0} u_j + \sum_{j:\mathbf{v}_{kj}=1} (1 - u_j) = |\mathbf{v}_k| + \sum_{j:\mathbf{v}_{kj}=0} u_j - \sum_{j:\mathbf{v}_{kj}=1} u_j \quad \forall k. \quad (1.16)$$

In this manner, every binary vector  $\mathbf{u} \in \mathbb{R}^{\lceil \log_2 m \rceil}$  having  $\sum_{j=1}^{\lceil \log_2 m \rceil} 2^{j-1} u_j \leq m-1$ , will have exactly one function  $A_k(\mathbf{u})$  equal to 0 and the remaining  $(m-1)$  such functions greater than or equal to 1. Finally, for each rectangle  $R_i$ , define a vector  $\mathbf{u}^i \in \mathbb{R}^{\lceil \log_2 m \rceil}$  of binary variables. Then [5] rewrites Problem P1 as Problem P2 below.

P2: minimize  $Y$

subject to (1.11), (1.12), (1.14)

$$y_i \leq L(k + (m-k)A_k(\mathbf{u}^i)) - \ell_i - s_i(w_i - \ell_i) \quad \forall (i, k), k \neq m \quad (1.17)$$

$$y_i \geq (k-1)L(1 - A_k(\mathbf{u}^i)) \quad \forall (i, k), k \neq 1 \quad (1.18)$$

$$\sum_{j=1}^{\lceil \log_2 m \rceil} 2^{j-1} u_j^i \leq m-1 \quad \forall i \quad (1.19)$$

$$x_i \geq 0, y_i \geq 0, \mathbf{u}^i \text{ binary } \forall i \quad (1.20)$$

Although unnoticed in [5], inequalities (1.17) are not needed when  $k = m$  because the  $m$  sheets of length  $L$  have a maximum height of  $mL$ , and inequalities (1.18) are not needed when  $k = 1$  because  $y_i \geq 0$  for all  $i$  by (1.20). Also, (1.19) is redundant for all  $i$  when  $\log_2 m = \lceil \log_2 m \rceil$ .

As alluded to above, there are two main differences between Problems P1 and P2. First, the sets  $S_{ij}$  of (1.7) found in P1 are replaced with the sets  $S'_{ij}$  of (1.14). Second, restrictions (1.8)–(1.10) and the binary variables  $Q_{ik}$  of P1 are replaced by (1.17)–(1.19) and the binary variables  $\mathbf{u}^i$ . For each  $i$ , and any chosen  $\mathbf{u}^i$  satisfying (1.19), that function  $A_k(\mathbf{u}^i)$  realizing value 0 will combine with inequalities (1.17) and (1.18) to enforce that

$$L(k-1) \leq y_i \leq Lk - \ell_i - s_i(w_i - \ell_i), \quad (1.21)$$

so that rectangle  $R_i$  is cut from sheet  $k$ . All  $2(m-1)$  remaining inequalities of (1.17) and (1.18)

for the chosen  $i$  are redundant. This same pair of inequalities is enforced in P2 when  $Q_{ik} = 1$ .

The paper [5] introduces Problem P2 as an improvement over P1. Problem P2 has  $n(2n + m - 1)$  constraints in (1.11),(1.12), (1.14) and (1.17)–(1.19),  $2n + 1$  continuous variables, and  $n(n + \lceil \log_2 m \rceil)$  binary variables. In contrast, and as noted above, Problem P1 has  $5n(n + 1)/2$  constraints,  $2n + 1$  continuous variables, and  $n(2n + m - 1)$  binary variables. Thus, Problem P2 has  $n(n + m - 1 - \lceil \log_2 m \rceil)$  fewer binary variables and  $(n(n + 7)/2) - 2m$  fewer constraints. (The paper [5] states that P1 has  $n(5n + 3)/2$  constraints and  $n(2n + m + 1)$  binary variables. The difference in the number of constraints is due to the omission of (1.10) from P1, but we have no explanation for the discrepancy in the number of variables. Also, the paper [5] states that P2 has  $n(2n + m + 1)$  constraints and  $n(n + \lceil \log_2 m \rceil)$  binary variables, with the difference in the number of constraints being that [5] includes the redundant inequalities (1.17) for  $k = m$  and (1.18) for  $k = 1$ .)

## Chapter 2

# Modeling Disjunctions

The two main modelling challenges identified in Section 1 can be viewed in terms of disjunctive programming. Given  $p$  polytopes of the form

$$\Omega_g \equiv \{\boldsymbol{\omega} : B^g \boldsymbol{\omega} \leq \mathbf{b}^g\} \quad \forall g = 1, \dots, p \quad (2.1)$$

in variables  $\boldsymbol{\omega}$ , each challenge is to construct linear inequalities that enforce

$$\boldsymbol{\omega} \in \bigcup_{g=1}^p \Omega_g. \quad (2.2)$$

Relative to the first challenge of preventing overlap between rectangles, for each  $(i, j) \in \mathcal{N}$ , statement (1.1) is of the form (2.2) with  $p = 4$ ,  $\boldsymbol{\omega} = (x_i, x_j, y_i, y_j, s_i, s_j)$ , and  $\Omega_g$  is antecedent  $g$  of (1.1) for  $g = 1, \dots, 4$ . For the second challenge of restricting that each rectangle  $R_i$  is cut from a single sheet, constraints (1.8)–(1.10) of P1 and (1.17)–(1.19) of P2 each enforce, in different ways, that (1.21) is satisfied for some  $k \in \{1, \dots, m\}$ . Given a fixed rectangle  $R_i$ , this restriction can be cast in the form of (2.2) by letting  $p = m$ ,  $\boldsymbol{\omega} = (y_i, s_i)$ , and

$$\Omega_g = \{(y_i, s_i) : y_i + \ell_i + s_i(w_i - \ell_i) \leq Lg, -y_i \leq -L(g - 1)\} \quad \forall g = 1, \dots, m. \quad (2.3)$$

In this section, we consider four different approaches for modelling (2.2), and then relate

these approaches in Section 3 to the 2-dimensional cutting stock problem. We let the index  $g$  run from 1 to  $p$  throughout.

- **Approach 1**

A classical method for modelling (2.2) is due to [2]. This paper defines  $p$  new sets of variables  $\boldsymbol{\mu}^g$  of the same size as  $\boldsymbol{\omega}$ , and  $p$  additional nonnegative variables, say  $\lambda_g$ , to form the set

$$\Omega' \equiv \left\{ (\boldsymbol{\omega}, \boldsymbol{\mu}, \boldsymbol{\lambda}) : B^g \boldsymbol{\mu}^g \leq \mathbf{b}^g \lambda_g \ \forall g, \ \boldsymbol{\omega} = \sum_{g=1}^p \boldsymbol{\mu}^g, \ 1 = \sum_{g=1}^p \lambda_g, \ \boldsymbol{\lambda} \geq \mathbf{0} \right\}, \quad (2.4)$$

where  $\boldsymbol{\mu}^T = ((\boldsymbol{\mu}^1)^T, \dots, (\boldsymbol{\mu}^p)^T)^T$ . The convex hull of  $\bigcup_{g=1}^p \Omega_g$  is shown to be the projection of the set  $\Omega'$  onto the space of the variables  $\boldsymbol{\omega}$ . When the sets  $\Omega_g$  lie within some larger optimization problem in  $\boldsymbol{\omega}$ , as is the case for the 2-dimensional cutting stock problem of concern, the convex hull is forfeited, and the variables  $\boldsymbol{\lambda}$  must be restricted to be binary to satisfy (2.2). A drawback of this approach is that  $\Omega'$  uses an additional  $p\eta$  variables  $\boldsymbol{\mu}$ , where  $\eta$  is the number of variables  $\boldsymbol{\omega}$ .

- **Approach 2**

A second approach for modelling (2.2) uses only the  $p$  binary variables  $\boldsymbol{\lambda}$  of Approach 1, in addition to the variables  $\boldsymbol{\omega}$ . For each  $g$ , define a vector  $\mathbf{U}^g \geq \mathbf{0}$  having the same size as  $\mathbf{b}^g$  in such a manner that the inequalities defining  $\Omega_g$  are redundant when the vector  $\mathbf{U}^g$  is added to  $\mathbf{b}^g$ ; that is, so that  $B^g \boldsymbol{\omega} \leq (\mathbf{b}^g + \mathbf{U}^g)$  for all  $\boldsymbol{\omega} \in \bigcup_{g=1}^p \Omega_g$ . Then define the system

$$\Gamma_1 \equiv \left\{ (\boldsymbol{\omega}, \boldsymbol{\lambda}) : B^g \boldsymbol{\omega} \leq \mathbf{b}^g + \mathbf{U}^g (1 - \lambda_g) \ \forall g, \ \sum_{g=1}^p \lambda_g = 1, \ \boldsymbol{\lambda} \text{ binary} \right\}. \quad (2.5)$$

That variable  $\lambda_g$  realizing value 1 in (2.5) will enforce that the associated system  $B^g \boldsymbol{\omega} \leq \mathbf{b}^g$  is satisfied, and the remaining  $p - 1$  systems will be redundant.

- **Approach 3**

A third approach for modelling (2.2) uses the vectors  $\mathbf{U}^g$  of Approach 2, as well as the

functions  $A_g(\mathbf{u})$  of (1.16), where  $m$  is replaced by  $p$  in (1.15) in defining the binary vectors  $\mathbf{v}_g \in \mathbb{R}^{\lceil \log_2 p \rceil}$  for all  $g$ . The property of the functions  $A_g(\mathbf{u})$  that, for any binary  $\mathbf{u} \in \mathbb{R}^{\lceil \log_2 p \rceil}$  with  $\sum_{j=1}^{\lceil \log_2 p \rceil} 2^{j-1} u_j \leq p-1$ , a single such function will equal to 0 and the remaining  $(p-1)$  functions will be greater than or equal to 1, allows us to conclude that  $\boldsymbol{\omega} \in \bigcup_{g=1}^p \Omega_g$  if and only if there exists a  $\mathbf{u}$  so that  $(\boldsymbol{\omega}, \mathbf{u}) \in \Gamma_2$ , with

$$\Gamma_2 \equiv \left\{ (\boldsymbol{\omega}, \mathbf{u}) : B^g \boldsymbol{\omega} \leq \mathbf{b}^g + \mathbf{U}^g A_g(\mathbf{u}) \ \forall g, \sum_{j=1}^{\lceil \log_2 p \rceil} 2^{j-1} u_j \leq p-1, \mathbf{u} \text{ binary} \right\}. \quad (2.6)$$

The inequality  $\sum_{j=1}^{\lceil \log_2 p \rceil} 2^{j-1} u_j \leq p-1$  is not needed when  $\log_2 p = \lceil \log_2 p \rceil$ .

- **Approach 4**

Approach 3 has an advantage over Approaches 1 and 2 in terms of numbers of binary variables, but this advantage can be overcome. Approach 3 uses only  $\lceil \log_2 p \rceil$  binary variables  $\mathbf{u}$ , while each of Approaches 1 and 2 uses  $p$  binary variables  $\boldsymbol{\lambda}$ . However, an observation of [1] shows that a restriction of the form  $\sum_{g=1}^p \lambda_g = 1$  in  $p$  binary variables  $\boldsymbol{\lambda}$  can be changed to having *continuous* variables  $\boldsymbol{\lambda} \geq \mathbf{0}$  by including the  $\lceil \log_2 p \rceil$  equations  $\sum_{g=1}^p \mathbf{v}_g \lambda_g = \mathbf{u}$  in binary variables  $\mathbf{u} \in \mathbb{R}^{\lceil \log_2 p \rceil}$ , where  $\mathbf{v}_g \in \mathbb{R}^{\lceil \log_2 p \rceil}$  for all  $g$ , as in Approach 3. This observation, together with a suitable projection operation, was used in [6] on the set  $\Omega'$  of Approach 1 to motivate linearizations of piecewise-linear functions. Applying this observation to Approach 2, we have that  $(\boldsymbol{\omega}, \boldsymbol{\lambda}) \in \Gamma_1$  if and only if there exists a  $\mathbf{u}$  so that  $(\boldsymbol{\omega}, \boldsymbol{\lambda}, \mathbf{u}) \in \Gamma_3$ , with

$$\Gamma_3 \equiv \left\{ (\boldsymbol{\omega}, \boldsymbol{\lambda}, \mathbf{u}) : B^g \boldsymbol{\omega} \leq \mathbf{b}^g + \mathbf{U}^g (1 - \lambda_g) \ \forall g, \sum_{g=1}^p \lambda_g = 1, \sum_{g=1}^p \mathbf{v}_g \lambda_g = \mathbf{u}, \boldsymbol{\lambda} \geq \mathbf{0}, \mathbf{u} \text{ binary} \right\}. \quad (2.7)$$

Then a vector  $\boldsymbol{\omega} \in \bigcup_{g=1}^p \Omega_g$  if and only if there exists a  $(\boldsymbol{\lambda}, \mathbf{u})$  so that  $(\boldsymbol{\omega}, \boldsymbol{\lambda}, \mathbf{u}) \in \Gamma_3$ .

A comparison of (2.6) and (2.7) reveals the following characteristics in terms of size and relaxation strength. Each of these two systems has  $\lceil \log_2 p \rceil$  binary variables  $\mathbf{u}$ , but (2.7) has an

additional  $p$  nonnegative variables  $\boldsymbol{\lambda}$  in  $\lceil \log_2 p \rceil + 1$  equations. Relative to strength, let  $\bar{\Gamma}_2$  denote the continuous relaxation of  $\Gamma_2$  obtained by replacing the  $\mathbf{u}$  binary restrictions with  $\mathbf{0} \leq \mathbf{u} \leq \mathbf{1}$ , and let  $\bar{\Gamma}_3$  denote the continuous relaxation of  $\Gamma_3$  obtained by deleting the  $\mathbf{u}$  binary restrictions. The paper [1, page 1483, Theorem 1] addresses the special family of disjunctions having  $\boldsymbol{\omega} \in \mathbb{R}^1$  and  $\Omega_g = \{\boldsymbol{\omega} : \boldsymbol{\omega} = \theta_g\}$  for some scalar  $\theta_g$  for each  $g$  and, for these disjunctions, shows that a simpler form of  $\bar{\Gamma}_3$  given by

$$\bar{\Gamma}_3 \equiv \left\{ (\boldsymbol{\omega}, \boldsymbol{\lambda}, \mathbf{u}) : \boldsymbol{\omega} = \sum_{g=1}^p \theta_g \boldsymbol{\lambda}_g, \sum_{g=1}^p \boldsymbol{\lambda}_g = \mathbf{1}, \sum_{g=1}^p \mathbf{v}_g \boldsymbol{\lambda}_g = \mathbf{u}, \boldsymbol{\lambda} \geq \mathbf{0} \right\}$$

is preferable to the method used to generate  $\bar{\Gamma}_2$  in terms of the strengths of the continuous relaxations. This relaxation preference with respect to  $\bar{\Gamma}_3$  over  $\bar{\Gamma}_2$  continues to hold true for the more general sets  $\Omega_g$  of (2.1), as stated in the following theorem.

### Theorem

Given any  $(\hat{\boldsymbol{\omega}}, \hat{\boldsymbol{\lambda}}, \hat{\mathbf{u}}) \in \bar{\Gamma}_3$  of (2.7), we have that  $(\hat{\boldsymbol{\omega}}, \hat{\mathbf{u}}) \in \bar{\Gamma}_2$  of (2.6).

### Proof

It is sufficient to show, given any  $(\hat{\boldsymbol{\lambda}}, \hat{\mathbf{u}})$  feasible to

$$P_1 \equiv \left\{ (\boldsymbol{\lambda}, \mathbf{u}) : \sum_{g=1}^p \boldsymbol{\lambda}_g = \mathbf{1}, \sum_{g=1}^p \mathbf{v}_g \boldsymbol{\lambda}_g = \mathbf{u}, \boldsymbol{\lambda} \geq \mathbf{0} \right\},$$

that  $\hat{\mathbf{u}}$  is feasible to

$$P_2 \equiv \left\{ \mathbf{u} : \sum_{g=1}^{\lceil \log_2 p \rceil} 2^{g-1} u_g \leq p - 1, \mathbf{0} \leq \mathbf{u} \leq \mathbf{1} \right\},$$

and that

$$(1 - \hat{\lambda}_g) \leq A_g(\hat{\mathbf{u}}) \forall g.$$

To begin, we have by definition that  $\mathbf{v}_g \in P_2$  for all  $g$ . Then  $\hat{\mathbf{u}} \in P_2$  because  $P_1$  expresses  $\hat{\mathbf{u}}$  as a



convex combination of the vectors  $\mathbf{v}_g$ . Next, the set  $P_1$  enforces that

$$(1 - \hat{\lambda}_g) = \sum_{\substack{j=1 \\ j \neq g}}^p \hat{\lambda}_j \leq \sum_{\substack{j=1 \\ j \neq g}}^p A_g(\mathbf{v}_j) \hat{\lambda}_j = \sum_{j=1}^p A_g(\mathbf{v}_j) \hat{\lambda}_j = A_g(\hat{\mathbf{u}}) \forall g.$$

To explain, consider any  $g$ . The first equality follows from  $\sum_{j=1}^p \hat{\lambda}_j = 1$ , the inequality is due to  $A_g(\mathbf{v}_j) \geq 1$  for all  $j \neq g$  and  $\hat{\boldsymbol{\lambda}} \geq \mathbf{0}$ , and the second equality is due to  $A_g(\mathbf{v}_g) = 0$ . The final equality is obtained from (1.16) by computing a linear combination of the equality constraints of  $P_1$  using the multiplier  $|\mathbf{v}_g|$  for the constraint  $\sum_{j=1}^p \hat{\lambda}_j = 1$  and, for each  $q = 1, \dots, \lceil \log_2 p \rceil$ , the multiplier  $\pi_q$  for constraint  $q$  of  $\sum_{j=1}^p \mathbf{v}_j \hat{\lambda}_j = \mathbf{u}$ , where  $\pi_q = 1$  if  $v_{gq} = 0$  and  $\pi_q = -1$  if  $v_{gq} = 1$ . In this manner, for each  $j = 1, \dots, p$ , we have  $|\mathbf{v}_j| + \boldsymbol{\pi}^T \mathbf{v}_j = |\mathbf{v}_j| + \sum_{q: v_{jq}=0} v_{jq} - \sum_{q: v_{jq}=1} v_{jq} = A_j(\mathbf{v}_g) = A_g(\mathbf{v}_j)$ , and we also have  $|\mathbf{v}_g| + \boldsymbol{\pi}^T \hat{\mathbf{u}} = A_g(\hat{\mathbf{u}})$ . Here,  $A_j(\mathbf{v}_g) = A_g(\mathbf{v}_j)$  for all  $j = 1, \dots, p$ , because the expression  $A_j(\mathbf{v}_g)$  can be interpreted as the Hamming distance between the binary vectors  $\mathbf{v}_g$  and  $\mathbf{v}_j$ ; that is, the number of positions at which the corresponding entries in  $\mathbf{v}_g$  and  $\mathbf{v}_j$  differ. This completes the proof.  $\square$

Consider the example below that shows that the converse of the theorem is not true. There can exist a  $(\hat{\boldsymbol{\omega}}, \hat{\mathbf{u}}) \in \bar{\Gamma}_2$  of (2.6) for which there exists no  $\hat{\boldsymbol{\lambda}}$  so that  $(\hat{\boldsymbol{\omega}}, \hat{\boldsymbol{\lambda}}, \hat{\mathbf{u}}) \in \bar{\Gamma}_3$  of (2.7).

### Example

Let  $p = 4$ ,  $\boldsymbol{\omega} \in \mathbb{R}^4$ , and  $\Omega_g = \{\boldsymbol{\omega} : 0 \leq \omega_g \leq 1, 0 \leq \omega_j \leq 2 \forall j \neq g\}$  for  $g = 1, \dots, 4$ . Then by defining, for each  $g$ ,  $\mathbf{U}^g \in \mathbb{R}^8$  to have entry 1 for the inequality  $\omega_g \leq 1$  and entry 0 for the inequalities  $\omega_j \leq 1 \forall j \neq g$  and  $-\omega_j \leq 0 \forall j$ , we have

$$\Gamma_2 = \{(\boldsymbol{\omega}, \mathbf{u}) : 0 \leq \omega_g \leq 1 + A_g(\mathbf{u}) \forall g = 1, \dots, 4, \mathbf{u} \text{ binary}\},$$

where  $\mathbf{u} \in \mathbb{R}^2$ ,  $A_1(\mathbf{u}) = u_1 + u_2$ ,  $A_2(\mathbf{u}) = (1 - u_1) + u_2$ ,  $A_3(\mathbf{u}) = u_1 + (1 - u_2)$ , and  $A_4(\mathbf{u}) =$

$(1 - u_1) + (1 - u_2)$ . For the same  $U^g$  and for  $\boldsymbol{\lambda} \in \mathbb{R}^4$ , system (2.7) can be expressed as

$$\Gamma_3 = \left\{ (\boldsymbol{\omega}, \boldsymbol{\lambda}, \mathbf{u}) : 0 \leq \omega_g \leq 1 + (1 - \lambda_g) \forall g = 1, \dots, 4, \sum_{g=1}^4 \lambda_g = 1, \right. \\ \left. \lambda_2 + \lambda_4 = u_1, \lambda_3 + \lambda_4 = u_2, \boldsymbol{\lambda} \geq \mathbf{0}, \mathbf{u} \text{ binary} \right\}.$$

The vector  $(\hat{\boldsymbol{\omega}}, \hat{\mathbf{u}})$  with  $\hat{\omega}_1 = \hat{\omega}_2 = \hat{\omega}_3 = \hat{\omega}_4 = 2$  and  $\hat{u}_1 = \hat{u}_2 = \frac{1}{2}$  is feasible to  $\bar{\Gamma}_2$ . Adding the sum of the four inequalities  $\omega_g \leq 1 + (1 - \lambda_g) \forall g$  to the equation  $-\sum_{g=1}^4 \lambda_g = -1$ , with all five restrictions from  $\bar{\Gamma}_3$ , gives  $\sum_{g=1}^4 \omega_g \leq 7$ , so that there exists no  $\hat{\boldsymbol{\lambda}}$  having  $(\hat{\boldsymbol{\omega}}, \hat{\boldsymbol{\lambda}}, \hat{\mathbf{u}}) \in \bar{\Gamma}_3$ .

We conclude this section with a modelling approach that is tailored for special sets  $\Omega_g$  of (2.1). This approach will be useful in dealing with the 2-dimensional cutting stock problem in the next section.

### Remark

Given that the sets  $\Omega_g$  of (2.1) have all  $B^g$  equal to the same matrix  $B$  so that

$$\Omega_g = \{\boldsymbol{\omega} : B\boldsymbol{\omega} \leq \mathbf{b}^g\} \forall g,$$

then disjunction (2.2) can be modelled by

$$B\boldsymbol{\omega} \leq \sum_{g=1}^p \mathbf{b}^g \lambda_g, \sum_{g=1}^p \lambda_g = 1, \boldsymbol{\lambda} \text{ binary}.$$

Given that there further exists some  $\bar{\mathbf{b}}$  so that

$$\mathbf{b}^g = (\mathbf{b}^1 - \bar{\mathbf{b}}) + \bar{\mathbf{b}}g \forall g, \tag{2.8}$$

the variables  $\boldsymbol{\lambda}$  can be replaced by a single integer variable, say  $z$ , to obtain

$$B\boldsymbol{\omega} \leq (\mathbf{b}^1 - \bar{\mathbf{b}}) + \bar{\mathbf{b}}z, 1 \leq z \leq p, z \text{ integer}. \tag{2.9}$$

This simplification follows because

$$\sum_{g=1}^p \mathbf{b}^g \lambda_g = \sum_{g=1}^p ((\mathbf{b}^1 - \bar{\mathbf{b}}) + \bar{\mathbf{b}}g) \lambda_g = (\mathbf{b}^1 - \bar{\mathbf{b}}) + \bar{\mathbf{b}} \left( \sum_{g=1}^p g \lambda_g \right) = (\mathbf{b}^1 - \bar{\mathbf{b}}) + \bar{\mathbf{b}}z,$$

where the first equation makes the substitution  $\mathbf{b}^g = (\mathbf{b}^1 - \bar{\mathbf{b}}) + \bar{\mathbf{b}}g$  for all  $g$  from (2.8), the second equation is due to  $\sum_{g=1}^p \lambda_g = 1$ , and the third equation is a standard binary expansion of the variable  $z$ .

# Chapter 3

## Proposed Models

We use the disjunctive programming arguments of the previous section to explain and enhance Problems P1 and P2 that were given in [5] to model the two-dimensional cutting stock problem. Recall that the two main modeling challenges are: preventing rectangle overlap and ensuring that each rectangle is cut from exactly one sheet. Each challenge is addressed below.

### Preventing Rectangle Overlap

The disjunctive statement (1.1) used to ensure that no two rectangles  $R_i$  and  $R_j$  are cut from the same region of any sheet can be expressed using any of Approaches 1, 2, 3, or 4. These approaches, and their relationships to the two methods of [5], are discussed below.

1. For each  $(i, j) \in \mathcal{N}$ , the implementation of Approach 1 of [2] requires the polyhedral sets  $\Omega_g$  of (2.1) to be bounded. Boundedness can be accomplished by including the twelve inequalities

$$0 \leq y_k \leq mL - \ell_k - s_k(w_k - \ell_k), \quad k = i, j, \quad (3.1)$$

$$0 \leq x_k \leq W - w_k - s_k(\ell_k - w_k), \quad k = i, j, \quad (3.2)$$

$$0 \leq s_k \leq 1, \quad k = i, j, \quad (3.3)$$

within each of the antecedents of (1.1). Then, letting  $p = 4$ , the sets  $\Omega_g$  for  $g = 1, \dots, 4$  of

(1.1) become as follows

$$\Omega_1 = \{(x_i, x_j, y_i, y_j, s_i, s_j) : x_i \geq x_j + w_j + s_j(\ell_j - w_j), (3.1), (3.2), (3.3)\}, \quad (3.4)$$

$$\Omega_2 = \{(x_i, x_j, y_i, y_j, s_i, s_j) : y_i \geq y_j + \ell_j + s_j(w_j - \ell_j), (3.1), (3.2), (3.3)\}, \quad (3.5)$$

$$\Omega_3 = \{(x_i, x_j, y_i, y_j, s_i, s_j) : x_i \leq x_j - w_i - s_i(\ell_i - w_i), (3.1), (3.2), (3.3)\}, \quad (3.6)$$

$$\Omega_4 = \{(x_i, x_j, y_i, y_j, s_i, s_j) : y_i \leq y_j - \ell_i - s_i(w_i - \ell_i), (3.1), (3.2), (3.3)\}. \quad (3.7)$$

Letting  $\boldsymbol{\omega} = (x_i, x_j, y_i, y_j, s_i, s_j)$  and  $\boldsymbol{\lambda} = (a_{ij}, b_{ij}, c_{ij}, d_{ij})$ , we can construct the set  $\Omega'$  of (2.4) in the six variables  $\boldsymbol{\omega}$ , 24 variables  $\boldsymbol{\mu}$ , and four variables  $\boldsymbol{\lambda}$ . The set  $\Omega'$  has 28 inequality restrictions, six equations of the form  $\boldsymbol{\omega} = \sum_{g=1}^4 \boldsymbol{\mu}^g$ , the equation  $a_{ij} + b_{ij} + c_{ij} + d_{ij} = 1$ , and nonnegativity on  $\boldsymbol{\mu}$ . Here, as noted earlier,  $a_{ij}, b_{ij}, c_{ij}, d_{ij}$  must also be binary to satisfy (2.2).

2. For each  $(i, j) \in \mathcal{N}$ , Approach 2 applied to (1.1) derives the set  $S_{ij}$  of Section 1, where (2.5) has  $p = 4$  and  $\boldsymbol{\omega} = (x_i, x_j, y_i, y_j, s_i, s_j)$ , and where  $\boldsymbol{\lambda}$  is given by  $(a_{ij}, b_{ij}, c_{ij}, d_{ij})$ . Each component of the vector  $\boldsymbol{U}^g \in \mathbb{R}^4$  has an entry of either  $W$  or  $mL$ , depending on whether the associated inequality restricts the variables  $(x_i, x_j)$  or  $(y_i, y_j)$ , respectively.
3. For each  $(i, j) \in \mathcal{N}$ , Approach 3 applied to (1.1) derives the set  $S'_{ij}$  of Section 1, using the same  $p = 4$ ,  $\boldsymbol{\omega} = (x_i, x_j, y_i, y_j, s_i, s_j)$ , and  $\boldsymbol{U}^g$  of  $S_{ij}$ . Unlike Approach 2, there are no variables  $\boldsymbol{\lambda}$ , and  $\boldsymbol{u} \in \mathbb{R}^2$  of (2.6) has  $u_1 = \alpha_{ij}$  and  $u_2 = \beta_{ij}$ . Expression (1.15) defines  $\boldsymbol{v}_1 = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$ ,  $\boldsymbol{v}_2 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$ ,  $\boldsymbol{v}_3 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$ , and  $\boldsymbol{v}_4 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$  so that (1.16) gives  $A_1(\boldsymbol{u}) = \alpha_{ij} + \beta_{ij}$ ,  $A_2(\boldsymbol{u}) = (1 - \alpha_{ij}) + \beta_{ij}$ ,  $A_3(\boldsymbol{u}) = \alpha_{ij} + (1 - \beta_{ij})$ , and  $A_4(\boldsymbol{u}) = (1 - \alpha_{ij}) + (1 - \beta_{ij})$ . The inequality  $\alpha_{ij} + 2\beta_{ij} \leq 3$  is not needed because  $\log_2 4 = \lceil \log_2 4 \rceil = 2$ .
4. For each  $(i, j) \in \mathcal{N}$ , Approach 4 applied to (1.1) derives the set

$$\begin{aligned} S''_{ij} = \{ & (x_i, x_j, y_i, y_j, s_i, s_j, a_{ij}, b_{ij}, c_{ij}, d_{ij}, \alpha_{ij}, \beta_{ij}) : (1.2) - (1.6), b_{ij} + d_{ij} = \alpha_{ij}, \\ & c_{ij} + d_{ij} = \beta_{ij}, a_{ij}, b_{ij}, c_{ij}, d_{ij} \geq 0, \alpha_{ij}, \beta_{ij} \text{ binary} \} \forall (i, j) \in \mathcal{N}. \end{aligned} \quad (3.8)$$

Here,  $\Gamma_3$  of (2.7) uses the same  $p = 4$ ,  $\boldsymbol{\omega} = (x_i, x_j, y_i, y_j, s_i, s_j)$ , and (continuous)  $\boldsymbol{\lambda} = (a_{ij}, b_{ij}, c_{ij}, d_{ij})$  of Approach 2, as well as the same vectors  $\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3$ , and  $\mathbf{v}_4$  of Approach 3 to compute the equations  $b_{ij} + d_{ij} = \alpha_{ij}$  and  $c_{ij} + d_{ij} = \beta_{ij}$  of (3.8), and also includes the variables  $\mathbf{u} \in \mathbb{R}^2$  given by  $u_1 = \alpha_{ij}$  and  $u_2 = \beta_{ij}$  as in Approach 3.

The Theorem and Example of Section 2 show that Approach 3 dominates that of Approach 2 relative to the relaxation strengths of the resulting sets. Consistent with the relaxations of  $\Gamma_2$  and  $\Gamma_3$  to  $\bar{\Gamma}_2$  and  $\bar{\Gamma}_3$ , respectively, let  $\bar{S}'_{ij}$  denote the continuous relaxations of  $S'_{ij}$  obtained by relaxing the  $\alpha_{ij}, \beta_{ij}$  binary restrictions to  $0 \leq \alpha_{ij}, \beta_{ij} \leq 1$ , and let  $\bar{S}''_{ij}$  denote the continuous relaxation of  $S''_{ij}$  obtained by deleting these same binary restrictions. For each  $(i, j) \in \mathcal{N}$ , the Theorem gives us that the set  $\bar{S}''_{ij}$  dominates  $\bar{S}'_{ij}$ . For an example of strict dominance relative to these specific sets, consider the trivial problem having  $m = 1$  sheet and  $n = 2$  rectangles  $R_1$  and  $R_2$ , with  $w_1 = w_2 = W$  and  $L_1 = L_2 = L$ . This problem cannot have a solution, as each rectangle is the same size as the sheet. But the point  $(x_1, x_2, y_1, y_2, s_1, s_2, \alpha_{12}, \beta_{12}) = (0, 0, 0, 0, 0, 0, \frac{1}{2}, \frac{1}{2})$  is feasible to  $\bar{S}'_{12}$ . However,  $\bar{S}''_{12}$  is empty, so that it identifies the problem as being infeasible. To explain, adding inequalities (1.2) and (1.3) to the negative of inequalities (1.4) and (1.5), with all four inequalities taken from (3.8), gives us  $W(a_{12} + c_{12}) + L(b_{12} + d_{12}) \leq 0$ . This last inequality cannot be satisfied because (1.6) of (3.8) enforces  $a_{12} + b_{12} + c_{12} + d_{12} = 1$ , and because  $a_{12}, b_{12}, c_{12}, d_{12} \geq 0$ .

### Ensuring that each Rectangle is Cut from One Sheet

Motivated by the disjunctive approaches of Section 2, and Problems P1 and P2 from [5], we present six models for ensuring that each rectangle is cut from exactly one sheet.

1. For each rectangle  $R_i$ , Approach 1 of [2] can be applied to a bounded version of the sets  $\Omega_g$  of (2.3) to model (2.2). Here,  $p = m$  within (2.2) and (2.4), and each set  $\Omega_g$  is of the form (2.3) with the two additional bounding inequalities  $0 \leq s_i \leq 1$ . Then  $\boldsymbol{\omega} = (y_i, s_i)$ ,  $\boldsymbol{\lambda} \in \mathbb{R}^m$ , and  $\boldsymbol{\mu} \in \mathbb{R}^{2m}$  with  $\boldsymbol{\mu}^g = (u_1^g, \mu_2^g)$  representing the products  $y_i \lambda_g$  and  $s_i \lambda_g$ , respectively, for

$g = 1, \dots, m$ . The resulting set  $\Omega'$  of (2.4) takes the form

$$\begin{aligned} \Omega' = \{ & \boldsymbol{\omega}, \boldsymbol{\mu}, \boldsymbol{\lambda} : L(g-1)\lambda_g \leq \mu_1^g \leq Lg\lambda_g - \ell_i\lambda_g - \mu_2^g(w_i - \ell_i) \forall g = 1, \dots, m, \\ & 0 \leq \mu_1^g \leq \lambda_g \forall g = 1, \dots, m, \\ & \sum_{g=1}^m \mu_1^g = y_i, \sum_{g=1}^m \mu_2^g = s_i, \sum_{g=1}^m \lambda_g = 1, \boldsymbol{\lambda} \geq \mathbf{0} \}. \end{aligned}$$

2. For each rectangle  $R_i$ , the sets  $\Omega_g$  of (2.3) satisfy the conditions of the Remark with  $\boldsymbol{\omega} = (y_i, s_i)$ , and with  $\mathbf{b}^1 = \begin{bmatrix} L \\ 0 \end{bmatrix}$  and  $\bar{\mathbf{b}} = \begin{bmatrix} L \\ -L \end{bmatrix}$  satisfying (2.8). Then (2.2) with  $p = m$  can be expressed in the form of (2.9) as

$$\bigcup_{g=1}^m \Omega_g = \{(y_i, s_i) : L(z_i - 1) \leq y_i \leq Lz_i - \ell_i - s_i(w_i - \ell_i), 1 \leq z_i \leq m, z_i \text{ integer}\}. \quad (3.9)$$

Within (3.9), the variable  $z_i$  identifies the sheet number from which rectangle  $R_i$  is to be cut.

3. For each rectangle  $R_i$ , binary expansions of the integer variable  $z_i$  found in (3.9) can be performed. One method is to define, for each  $R_i$ ,  $m$  new binary variables, say  $Q_{ik}$  for  $k = 1, \dots, m$ , and to set

$$z_i = \sum_{k=1}^m Q_{ik}k, \sum_{k=1}^m Q_{ik} = 1, Q_{ik} \text{ binary } \forall k = 1, \dots, m. \quad (3.10)$$

This method yields restrictions (1.8)–(1.10) of Problem P1, upon observing that  $\sum_{k=1}^m Q_{ik}k - 1 = \sum_{k=1}^m Q_{ik}(k-1)$  for each  $i$ . An alternate method is to define, for each rectangle  $R_i$ , only  $\lceil \log_2 m \rceil$  binary variables  $Q_{ik}$  for  $k = 1, \dots, \lceil \log_2 m \rceil$ , and to set

$$z_i = \sum_{k=1}^{\lceil \log_2 m \rceil} 2^{k-1} Q_{ik}, \sum_{k=1}^{\lceil \log_2 m \rceil} 2^{k-1} Q_{ik} \leq m, Q_{ik} \text{ binary } \forall k = 1, \dots, \lceil \log_2 m \rceil. \quad (3.11)$$

This third method shares the advantage of Approaches 3 and 4 that a logarithmic number of binary variables is needed for each family of disjunctions.

4. For each rectangle  $R_i$ , Approach 2 with  $p = m$  is applicable to (2.3), where each set  $\Omega_g$  has

$\mathbf{U}^g \in \mathbb{R}^2$  in (2.5) so that the upper bounds on the left and right inequalities of (2.3) are  $(m-g)L$  and  $(g-1)L$ , respectively. In this manner, we obtain

$$\bigcup_{g=1}^m \Omega_g = \left\{ (y_i, s_i, \boldsymbol{\lambda}) : y_i + \ell_i + s_i(w_i - \ell_i) \leq Lg + (m-g)L(1 - \lambda_g) \forall g = 1, \dots, m, \right. \\ \left. -y_i \leq -L(g-1) + L(g-1)(1 - \lambda_g), \sum_{g=1}^m \lambda_g = 1, \boldsymbol{\lambda} \text{ binary} \right\}. \quad (3.12)$$

Here, the first inequality is not needed when  $g = m$ , given that there exists a sufficient number  $m$  of sheets to cut the rectangles, and the second inequality is not needed when  $g = 1$ , assuming that  $y_i \geq 0$  is elsewhere enforced.

5. For each rectangle  $R_i$ , the application of Approach 3 with  $p = m$  to (2.3) uses the same upper bounds  $\mathbf{U}^g \in \mathbb{R}^2$  within (2.6) as Approach 2 does within (2.5), as discussed above. The functions  $A_g(\mathbf{u})$  are defined as in (1.15) in terms of the vectors  $\mathbf{v}_g$  given in (1.16). Here, only  $\lceil \log_2 m \rceil$  binary variables are required for each  $R_i$ . Upon considering all rectangles  $R_i, i = 1, \dots, m$ , inequalities (1.17)–(1.19) of Problem P2 result.
6. For each rectangle  $R_i$ , Approach 4 with  $p = m$  is applicable to (2.3) using the same upper bounds  $\mathbf{U}^g \in \mathbb{R}^2$  in (2.7) as Approaches 2 and 3 within (2.5) and (2.6), respectively, and using the same vectors  $\mathbf{v}_g \in \mathbb{R}^{\lceil \log_2 m \rceil}$  defined in (1.15) as Approach 3. The resulting model has  $m$  continuous variables  $\boldsymbol{\lambda}$  and  $\lceil \log_2 m \rceil$  binary variables  $\mathbf{u}$ , as given below.

$$\bigcup_{g=1}^m \Omega_g = \left\{ (y_i, s_i, \boldsymbol{\lambda}) : y_i + \ell_i + s_i(w_i - \ell_i) \leq Lg + (m-g)L(1 - \lambda_g) \forall g = 1, \dots, m, \right. \\ \left. -y_i \leq -L(g-1) + L(g-1)(1 - \lambda_g), \sum_{g=1}^m \lambda_g = 1, \right. \\ \left. \sum_{g=1}^m \mathbf{v}_g \lambda_g = \mathbf{u}, \boldsymbol{\lambda} \geq \mathbf{0}, \mathbf{u} \text{ binary} \right\}. \quad (3.13)$$

Again, as with Approaches 2 and 3, the first inequality is not needed when  $g = m$ , and the second inequality is not needed when  $g = 1$ .



## Chapter 4

# Computational Experience

The previous section presented four models for preventing rectangle overlap and six models for ensuring that each rectangle is cut from a single sheet. We provide computational experience in this section to compare the merits of different pairwise-combinations of these models, focusing on initial relaxation value, numbers of nodes enumerated in a branch-and-bound routine, and overall CPU execution times.

Our formulations can be viewed as compartmentalized in that, in addition to a common objective function and set of constraints, the restrictions for preventing rectangle overlap and the restrictions for ensuring that each rectangle is cut from a single sheet are substitutable. For all problems, the objective is to minimize  $Y$ , as in Problems P1 and P2, and the common constraints are (1.11), (1.12), and the nonnegativity restrictions  $x_i \geq 0, y_i \geq 0 \forall i$  of (1.13). For enforcing the prevention of rectangle overlap, Model 1 forms a set  $\Omega'$  of (2.4) comprised of  $\Omega_1$  through  $\Omega_4$  of (3.4) through (3.7), respectively, for each  $(i, j), i < j$ . In lieu of these restrictions, Models 2, 3, and 4 enforce the set  $S_{ij}$  of Section 1, the set  $S'_{ij}$  of Section 1, and the set  $S''_{ij}$  of Section 3 as defined in (3.8), respectively. As mentioned earlier, the variable  $s_i$  is not necessary for any rectangle  $R_i$  which is a square (i.e.  $l_i = w_i$ ) and is thus set to 0. Relative to ensuring that each rectangle  $R_i$  is cut from a single sheet, for each  $i = 1, \dots, m$ , Method 2 uses the integer variable  $z_i$  and inequalities as described in (3.9). Method 3 uses binary expansions of the integer variables  $z_i$  as presented in (3.10) to obtain restrictions of the form (1.8)–(1.10). (We chose to not perform

base-2 expansions on the variables  $z_i$  as in (3.11) because preliminary computational experience did not indicate an advantage.) Method 4 uses the inequalities of (3.12), while Method 5 uses the restrictions (1.17)–(1.19) of P2, together with  $\mathbf{u}^i$  binary for all  $i$ . Method 6 employs the sets described in (3.13).

All test problems were coded in AMPL and submitted to CPLEX 12.6.0.0 on Clemson University’s Palmetto cluster (a 21,392-core 500 teraFLOPS High Performance Computing System), using one of the cluster’s Intel-based cores, with 24gb of RAM.

We begin by examining the two sample problems of [5], which we refer to as Sample Problems 1 and 2, and abbreviate by SP1 and SP2, respectively. Problem SP1 has  $n = 7$  rectangles and  $m = 4$  sheets, while SP2 has  $n = 18$  rectangles and  $m = 4$  sheets. Specific rectangle and sheet dimensions for each problem are found in Appendix A. For both of these problems, we paired each of the four models for preventing rectangle overlap with each of models 2 through 6 for ensuring that every rectangle is cut from a single sheet. (Method 1 was not used for reasons discussed later.) Table 1 gives the results for SP1, and Table 2 for SP2. Each table contains six columns. The first column is an ordered pair whose first entry denotes the model type used to prevent rectangle overlap and whose second entry denotes the model type used to ensure that each rectangle is cut from a single sheet. In this manner, Problems P1 and P2 of Section 1 are represented in the first column by ordered pairs (2,3) and (3,5) respectively. Columns 2 and 3 denote the objective values to the mixed-integer programs, and to the associated linear programs obtained by relaxing the discrete variables to be continuous, respectively. Columns 4, 5, and 6 give the number of simplex iterations, the number of branch-and-bound nodes encountered, and the overall CPU execution time in seconds for CPLEX 12.6.0.0 to solve the problem.

Some observations follow from Tables 1 and 2. First, these tables indicate, for both test problems, that the linear programming relaxations of the first set of five problem forms are tighter than those available from all other forms, but the number of MIP simplex iterations, branch-and-bound nodes, and CPU times are much larger. This increased effort is likely due to the larger formulation sizes resulting from the approach of [2]. (Based on this performance, we did not use Method 1 to ensure that each rectangle is cut from one sheet.) Second, Method 4 for preventing

rectangle overlap appears inferior to both Methods 2 and 3, as can be seen from columns 4 through 6 of the tables. This result is unexpected, as Method 4 uses half the number of binary variables as does Method 2. Third, Methods 2 and 3 for preventing rectangle overlap are comparable, although both appear to suffer when combined with Method 6 for ensuring that each rectangle is cut from one sheet. This is consistent with earlier results, since Method 4 for preventing rectangle overlap and Method 6 for cutting from a single sheet both use Approach 4 of Section 2 for handling disjunctions. Fourth, for both problems tested, Problem P1 (form (2,3)) outperformed Problem P2 (form (3.5)), which contradicts the experience of [5]. Finally, these two problems appear to be easier to solve than other problems of similar and slightly larger sizes, as we were unable to solve various problems having from 16 to 39 rectangles and 4 to 11 sheets in 10800 seconds CPU execution time.

In order to isolate the effects of the four different modeling forms for preventing rectangle overlap, we reconsider the two sample problems of [5], but this time consider only a single sheet having length  $L = \sum_{i=1}^n \ell_i$  so that it is sufficient to contain all rectangles, and we remove all constraints associated with multiple sheets. We refer to these problems as SP1' and SP2', respectively, to distinguish them from SP1 and SP2. The results are found in Tables 3 and 4. For each problem, we considered all four forms, as indicated in the first column. The remaining columns are identical to those found in Tables 1 and 2. Again, Method 1 is superior in terms of the tightness of the linear programming relaxations, but is not competitive in terms of the numbers of MIP simplex iterations, branch-and-bound nodes, and CPU times. Methods 2, 3, and 4 are competitive, though Method 2 appears preferable. As with the case of multiple sheets, this result is unexpected, since Method 2 uses twice the number of binary variables as do Methods 3 and 4.

Table 4.1: SP1 ( $n = 7$  rectangles,  $m = 4$  sheets)

Problem Form	Objective Value	LP Relaxation	MIP simplex iterations	B&B nodes	CPU time (seconds)
(1,2)	277	100.00	39288	1137	2.04
(1,3)	277	100.00	28987	967	1.94
(1,4)	277	100.00	18235	753	1.46
(1,5)	277	100.00	36955	920	2.03
(1,6)	277	100.00	58610	1766	3.37
(2,2)	277	82.00	2906	447	0.50
(2,3)	277	82.00	825	169	0.35
(2,4)	277	82.00	1235	233	0.27
(2,5)	277	82.00	8029	1399	0.63
(2,6)	277	82.00	15981	2821	6.15
(3,2)	277	82.00	11501	3676	0.86
(3,3)	277	82.00	2314	625	0.62
(3,4)	277	82.00	6364	2067	0.73
(3,5)	277	82.00	12641	3126	0.88
(3,6)	277	82.00	8195	2572	8.36
(4,2)	277	82.00	21013	6077	1.26
(4,3)	277	82.00	9786	2538	0.88
(4,4)	277	82.00	13141	3932	1.00
(4,5)	277	82.00	23844	5863	1.37
(4,6)	277	82.00	31586	6015	6.87

Table 4.2: SP2 ( $n = 18$  rectangles,  $m = 4$  sheets)

Problem Form	Objective Value	LP Relaxation	MIP simplex iterations	B&B nodes	CPU time (seconds)
(1,2)	525	146.28	20052284	107683	6454.59
(1,3)	525	146.28	34960168	426967	9551.93
(1,4)	525	146.28	184213342	1645024	41568.30
(1,5)	525	146.28	22227939	63699	7334.74
(1,6)	525	146.28	176751280	520508	77295.40
(2,2)	525	100.00	279663	16718	29.09
(2,3)	525	100.00	168372	7953	19.58
(2,4)	525	100.00	248573	18375	21.94
(2,5)	525	100.00	634316	20044	45.03
(2,6)	525	100.00	293006	25333	29.18
(3,2)	525	100.00	223864	33165	27.36
(3,3)	525	100.00	208126	29178	24.72
(3,4)	525	100.00	304061	40545	35.29
(3,5)	525	100.00	301985	34846	39.00
(3,6)	525	100.00	407499	47532	56.25
(4,2)	525	100.00	1725174	186727	152.31
(4,3)	525	100.00	1015173	130450	98.11
(4,4)	525	100.00	1997947	224633	190.65
(4,5)	525	100.00	762133	93993	127.03
(4,6)	525	100.00	503079	62109	54.99

Table 4.3: SP1' ( $n = 7$  rectangles,  $m = 1$  sheet)

Problem Form	Objective Value	LP Relaxation	MIP simplex iterations	B&B nodes	CPU time (seconds)
1	272	107.35	376820	10381	18.62
2	272	82.00	44629	4865	2.23
3	272	82.00	85422	12116	3.84
4	272	82.00	157400	25967	8.16

Table 4.4: SP2' ( $n = 18$  rectangles,  $m = 1$  sheet)

Problem Form	Objective Value	LP Relaxation	MIP simplex iterations	B&B nodes	CPU time (seconds)
1	525	146.28	22051429	99842	8245.21
2	525	100.00	487113	21981	52.13
3	525	100.00	700183	96610	93.26
4	525	100.00	633292	97040	67.23

## Chapter 5

# Conclusions and Discussion

This paper presents a general disjunctive framework for modeling a special family of 2-dimensional cutting stock problems. Two main disjunctive challenges exist: preventing rectangle overlap and ensuring that each rectangle is cut from a single sheet. Four approaches for modeling unions of polytopes are reviewed; these four approaches give rise to four models for preventing rectangle overlap and six models for ensuring that each rectangle is cut from a single sheet. The models contain different numbers of discrete and continuous variables, different numbers of constraints, and promote different relaxation values. The models were submitted to CPLEX 12.6.0.0 for the purpose of comparing their relative merits.

A surprising outcome of the computational results is the inherent difficulty of the problem. The larger of the two test problems taken from the literature contains only  $n = 18$  rectangles and  $m = 4$  sheets. We were unable to solve slightly larger problems within 1800 CPU seconds by any of the models described.

Based on insights gained from this study, we have identified three avenues for future work. First, the problems suffer from symmetry, which can slow the enumerative process. To illustrate, consider any arrangement of rectangles upon a single sheet having width  $W$  and length  $L$ , with lower left corner placed at the origin and with width parallel to the  $x$ -axis. This arrangement can be reflected about the vertical line  $x = \frac{W}{2}$ , or about the horizontal line  $y = \frac{L}{2}$ , or about both the vertical and horizontal lines. Thus, four different configurations describe similar cutting patterns.

Given a cutting pattern on  $m$  sheets, there consequently exists  $4^m$  different (equivalent) patterns. The manner in which the sheets are numbered increases this value to  $(4^m)!$ . The challenge, and first avenue of research, is to avoid this symmetry. Notably, the CPLEX option for elimination of symmetry had no effect on iteration counts or solve times. The second avenue of research is to compute improved disjunctions that tighten the linear programming relaxations, but do not substantially increase the problem sizes. Tables 1 through 4 indicate significant gaps between the integer optimal solutions and the linear programming relaxation values, particularly when Approach 1 is not used. The third research direction is to exploit the linear programming strength afforded by Approach 1 through the identification of special structures. Our ongoing research addresses these avenues.



## Appendix: Cutting Rectangle and Sheet Dimensions

Problem Name	Sheet Dimensions	Set of Rectangle Dimensions $(\ell_i, w_i)$
SP1	110×60	(82,60),(90,30),(85,27),(57,30) (60,25), (60,20), (55,29)
SP2	180×150	(130,30),(130,10),(120,25),(100,100) (95,95),(90,90),(95,85),(80,80) (80,75),(70,70),(60,60),(55,50) (40,40),(50,40),(100,30),(45,20) (20,15),(25,10)
SP1'	489×60	same as SP1
SP2'	1385×150	same as SP2

# Bibliography

- [1] Adams, W.P. and Henry, S.M., Base-2 Expansions for Linearizing Products of Functions of Discrete Variables, *Operations Research*, Vol. 60, No. 6, 1477–1490, 2012.
- [2] Balas, E. “Disjunctive Programming and a Hierarchy of Relaxations for Discrete Optimization Problems,” *SIAM Journal on Algebraic and Discrete Methods*, Vol. 6, No. 3, 466–486, 1985.
- [3] Chen, C.-S., Sarin, S., and Ram, B., A Mixed-Integer Programming Model for a Class of Assortment Problems, *European Journal of Operational Research*, Vol. 65, No. 3, 362–367, 1993.
- [4] Li, H.L. and Chang, C.T., An Approximately Global Optimization Method for Assortment Problems, *European Journal of Operational Research*, Vol. 105, No. 3, 604–612, 1998.
- [5] Lu, H.-C., Ko, Y.-C., and Huang, Y.-H., A Note on ‘Reducing the Number of Binary Variables in Cutting Stock Problems’, *Optimization Letters*, Vol. 8, No. 2, pp 569–579, 2014.
- [6] Muldoon, F., Polyhedral Approximations of Quadratic Semi-Assignment Problems, Disjunctive Programs, and Base-2 Expansions of Integer Variables, Ph.D. Dissertation, Clemson University, 2012.