Implementation of $\mathcal{V}$-Polyhedral Disjunctive Cuts in SCIP

vorgelegt von
Gennesaret Kharistio Tjusila
Berlin

der Fakultät II - Mathematik und Naturwissenschaften
der Technische Universität Berlin
zur Erlangung des akademisches Grades

Master Of Science
- M. Sc. -

Wissenschaftliche Betreuung:	Dr. Mathieu Besançon
	Université Grenoble Alpes, Inria, LIG
Prüfer:	Priv. Doz. Dr. Timo Berthold
	Technische Universität Berlin
	Prof. Dr. Thorsten Koch
	Technische Universität Berlin

Berlin, den July 8, 2025

Eigenständigkeitserklärung

Hiermit versichere ich, dass ich die vorliegende Arbeit eigenständig ohne Hilfe Dritter und ausschließlich unter Verwendung der aufgeführten Quellen und Hilfsmittel angefertigt habe. Alle Stellen die den benutzten Quellen und Hilfsmitteln unverändert oder sinngemäß entnommen sind, habe ich als solche kenntlich gemacht.

Sofern generative KI-Tools verwendet wurden, habe ich Produktnamen, Hersteller, die jeweils verwendete Softwareversion und die jeweiligen Einsatzzwecke (z.B. sprachliche Überprüfung und Verbesserung der Texte, systematische Recherche) benannt. Ich verantworte die Auswahl, die Übernahme und sämtliche Ergebnisse des von mir verwendeten KI-generierten Outputs vollumfänglich selbst.

Die Satzung zur Sicherung guter wissenschaftlicher Praxis an der TU Berlin vom 15. Februar 2023. `https://www.static.tu.berlin/fileadmin/www/10002457/K3-AMBl/Amtsblatt_2023/Amtliches_Mitteilungsblatt_Nr._16_vom_30.05.2023.pdf` habe ich zur Kenntnis genommen.

Ich erkläre weiterhin, dass ich die Arbeit in gleicher oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegt habe.

Berlin, den July 8, 2025

Gennesaret Kharistio Tjusila

Erklärung zur Nutzung Künstlicher Intelligenz

ChatGPT o3 und 4o wurden bei der Erstellung dieser Arbeit als Schreibassistenz eingesetzt, insbesondere zur Unterstützung bei der Grammatikprüfung, beim Umformulieren von Texten sowie als Referenz bei der Programmierung. GitHub Copilot wurde zur Unterstützung beim Schreiben des im Rahmen dieser Arbeit verwendeten Codes eingesetzt.

Sämtliche finalen Texte und Quellcodes wurden vom Verfasser überprüft und eigenständig validiert. Dabei wurde sichergestellt, dass alle von den genannten Tools bereitgestellten Ideen und Vorschläge — soweit übernommen — ordnungsgemäß zitiert oder entsprechend den wissenschaftlichen Standards integriert wurden. Der Verfasser übernimmt die volle Verantwortung für alle Inhalte dieser Arbeit.

Berlin, den July 8, 2025

Gennesaret Kharistio Tjusila

Zusammenfassung

V-polyhedral disjunctive cuts (VPC) sind ein Algorithmus zur Erzeugung von Schnittebenen aus gültigen disjunktiven Mengen – Vereinigungen paarweise disjunkter Polyeder, deren abgeschlossene konvexe Hülle die ganzzahlige Hülle eines gemischt-ganzzahligen linearen Programms enthält, wie sie beispielsweise an den Blättern eines partiellen Branch-and-Bound-Baums entstehen. Der Algorithmus basiert auf der Lösung eines Hilfs-LP, dessen Eingabe aus der V-Polyhedral Darstellung der Elemente der disjunktiven Menge stammt. Da die V-Polyhedral Darstellung exponentiell groß werden kann, verwenden praktische Implementierungen des Algorithmus stattdessen Corner-Polyeder-Relaxationen der Elemente, da diese eine einfache V-Polyhedral Darstellung besitzen.

In dieser Arbeit stellen wir den Algorithmus vor und erweitern die bestehende Theorie. Insbesondere erläutern wir ein Verfahren zur Konstruktion der Corner-Polyeder-Relaxationen, das sicherstellt, dass das Hilfs-LP stets lösbar ist. Anschließend implementieren wir den Algorithmus in SCIP und testen die resultierende Implementierung anhand des MIPLIB2017-Benchmark-Sets. Unsere numerischen Experimente konzentrieren sich auf die Analyse der closed root node gap sowie auf den Einfluss der erzeugten Schnittebenen auf den Branch-and-Bound-Prozess. In unseren root node gap-Experimenten bestätigen wir frühere Ergebnisse, die mit einem anderen Löser und einem teilweise überlappenden Instanzenset erzielt wurden. Diese zeigen, dass VPC allein in der Mehrheit der Instanzen weniger root node gap schließt als der Gomory Mixed Integer Cut. Wird VPC jedoch gemeinsam mit anderen in bestehenden Lösern implementierten Schnittebenen eingesetzt, ergänzt es die vorhandenen Schnittebenenverfahren und schließt – verglichen mit der alleinigen Verwendung der Standardverfahren – über die Menge der anwendbaren Instanzen hinweg mehr gap. Wir bestätigen außerdem frühere Beobachtungen, dass VPC tendenziell besser bei Instanzen mit ausschließlich binären Variablen funktioniert als bei solchen mit allgemeinen Ganzzahlen (hinsichtlich der geschlossenen root node gap).

Unsere Experimente zum Branch-and-Bound-Verfahren liefern hingegen weniger eindeutige Ergebnisse. Selbst wenn VPC ohne zusätzliche Kosten eingesetzt wird, zeigt sich keine Verbesserung im Branch-and-Bound-Prozess. Daher schließen wir, dass weitere Forschung notwendig ist, um zu verstehen, unter welchen Bedingungen VPC so eingesetzt werden kann, dass es den Branch-and-Bound-Prozess tatsächlich verbessert.

Abstract

V-polyhedral disjunctive cut (VPC) is an algorithm to generate cutting planes from valid disjunctive sets—unions of pairwise disjoint polyhedra whose closure of the convex hull contains the integer hull of a mixed-integer linear program, such as the leaves of a partial branch-and-bound tree. The algorithm is based on solving an auxiliary LP whose input comes from the V-polyhedral representation of the elements of the disjunctive set. Since the V-polyhedral description can become exponentially large, practical implementations of the algorithm use corner polyhedron relaxations of the elements instead, as these admit a simple V-polyhedral representation.

In this thesis, we give a presentation of the algorithm and extend the existing theory. In particular, we explain a scheme for constructing corner polyhedron relaxations such that the auxiliary LP is always feasible. Next, we implement the algorithm in SCIP and test the resulting implementation on the MIPLIB2017 benchmark set. Our computational study focuses on evaluating the root node gap closed and the impact of the generated cutting planes on the branch-and-bound process. In our root node gap closed experiments, we reinforce previous results obtained with another solver and a partially overlapping instance set. These results show that although VPC alone closes less root node gap in the majority of instances compared to the Gomory Mixed Integer Cut, when VPC is used in conjunction with other cutting planes already implemented in existing solvers, it augments the existing cutting plane set and closes more gap—compared to using only the default cuts—across the subset of instances where the cut generation algorithm is applicable. We also reinforce previous findings that VPC tends to perform better on instances where the integer variables consist only of binaries rather than general integers (in terms of closing more root node gap).

Our branch-and-bound experiments are less conclusive. We observe that, even when given for free, VPC does not improve the performance of the branch-and-bound process. Hence, we conclude that further research is necessary to study when VPC can be applied in a way that benefits the branch-and-bound process.

Acknowledgements

As I review the final draft of this thesis, I find myself looking back at the acknowledgments in my bachelor thesis. Back then, I began by saying that writing a thesis is fun — and in the condition I'm in now, I'm not sure I can say the same. What I do know, though, is that I have learned a lot, and this would never have been such a fruitful experience without the people around me.

First of all, I would like to thank my parents for their support throughout my studies. I thank Mathieu Besançon for first suggesting this topic and for supporting me all the way to the finish line, and Timo Berthold for being willing to co-supervise and serve as the first examiner of this work.

I thank my colleagues at ZIB, who were always more than happy to answer my questions — or simply hear my rants: Mohammed Ghannam, Marouane Felloussi, and Mark Turner. Most of all, I thank my boss, Gioni Mexi, for allowing me to work on this during my time at ZIB.

I'm grateful to my fellow students, Michael Adipoetra and Arvell Leoputra, for helping me proofread this work and telling me when I wasn't making sense.

Finally, I thank my non-math friends, Georgius Kenneth and Stefanus Theo, for always being there to support me.

Contents

Introduction

A mixed-integer linear program (MIP) is the problem of minimizing a linear function subject to a system of linear inequalities and integrality constraints. Most state-of-the-art solvers, such as SCIP [1] and XPress [2], implement the branch-and-bound method to solve MIPs. It has been shown in the literature that the addition of cutting planes—linear inequalities satisfied by all feasible integer points of the MIP but violated by some fractional solutions of the LP relaxation—improves the performance of branch-and-bound solvers [3]. This thesis explores a recently proposed algorithm for generating cutting planes, known as $\mathcal{V}$-polyhedral disjunctive cuts [4].

The key idea behind $\mathcal{V}$-polyhedral disjunctive cuts is to generate cuts from valid *disjunctive sets*—unions of pairwise disjoint polyhedra whose closure of the convex hull contains the feasible set of the MIP. It does so by constructing an auxiliary LP whose inputs are $\mathcal{V}$-polyhedral representations of the polyhedra in the disjunctive set. Since the $\mathcal{V}$-polyhedral representation may be exponential in size compared to the linear inequality description of the disjunctive set, a relaxation scheme based on corner polyhedron relaxations is used.

The central question of this thesis is whether the results of [4] can be reproduced using different solver settings and a standard benchmark instance set such as MIPLIB2017 [5]. This motivation stems from several limitations of the original work. First, the algorithm in [4] is implemented on top of Cbc [6], a solver known to be inferior to SCIP in terms of robustness and performance (see, for example, [7]). Second, the reported branch-and-bound results rely on Gurobi [8], a commercial solver that does not provide transparent access to the internals of the solving process, limiting insight into the role of the cuts. Finally, the computational experiments in [4] are conducted on a combined set of instances drawn from various sources, rather than a single benchmark set.

To address these limitations, we implement the algorithm within SCIP, an open-source solver developed at the Zuse Institute Berlin. This thesis makes several contributions. First, we extend the theoretical insights of [4] in two ways: we expand the details of several proofs that were only sketched in the original work, and we propose a new method for generating corner polyhedron relaxations, which guarantees feasibility of the auxiliary LPs used during cut generation. Second, we reimplement the algorithm of [4] within SCIP. Finally, we present computational results. We run root node gap experiments and full branch-and-bound runs on MIPLIB2017 instances. Our root node results confirm that VPCs are particularly effective for instances of lower dimension (defined as the sum of rows and columns). This is noteworthy because small instance dimension is used as a selection criterion in [4]. They also tend to perform better in pure binary MIPs than in models with general integer variables. Finally, we find that while VPCs alone are weaker than Gomory mixed-integer cuts, they provide a complementary strengthening effect when used in conjunction with SCIP's default cuts. The last two findings reinforce conclusions from [4]. However, our full branch-and-bound experiments show a different picture: on average, the application of VPCs degrades solver performance—even when cuts are provided "for free," that is, without accounting for their generation time.

The remainder of this thesis is organized as follows. In Chapter 1, we present the mathematical background necessary to understand the rest of the thesis. We focus in particular on the branch-and-bound method for solving MIPs, the simplex method for solving linear programs, and basic results from polyhedral theory. We also present the corner polyhedron relaxation.

In Chapter 2, we introduce the theory of $\mathcal{V}$-polyhedral disjunctive cuts. We begin with a preliminary note on cutting planes and disjunctive sets. Then, we give an in-depth treatment of the theory of $\mathcal{V}$-polyhedral disjunctive cuts. We expand on several of the proofs from [4] and propose a new method for ensuring the feasibility of the auxiliary LP used in the generation of

$\mathcal{V}$-polyhedral disjunctive cuts.

In Chapter 3, we describe several aspects of our implementation of the algorithm within SCIP. This chapter is written to give readers insights into the process of implementing theoretical cutting planes into solvers.

Finally, in Chapter 4, we present our computational results. We begin with root node gap closed experiments, followed by a branch-and-bound comparison.

Chapter 1

Mathematical Preliminaries

We begin with a review of background knowledge in mixed-integer linear programs that is necessary to understand this thesis. This also serves as an introduction to the notation that will be used throughout. Section 1.1 presents mixed-integer linear programs and related terminology. Section 1.2 discusses the branch-and-bound method for solving them. The simplex method for linear programs is covered in Section 1.3. In Section 1.4, we introduce the dual representation theorem for polyhedra. Finally, we introduce the corner polyhedron in Section 1.5.

1.1 Mixed-Integer Linear Programs

Throughout this thesis, we use the notation $[n] := \{1, \dots, n\}$ where $n \in \mathbb{N}$.

Definition 1.1. Let $m, n \in \mathbb{Z}$. Given a *cost vector* $c \in \mathbb{R}^n$, a *constraint matrix* $A \in \mathbb{R}^{m \times n}$, a left-hand and right-hand side vector $L \in (\mathbb{R} \cup \{-\infty\})^m$, $U \in (\mathbb{R} \cup \{+\infty\})^m$, a lower and upper bound vector $\ell \in (\mathbb{R} \cup \{-\infty\})^n$, $u \in (\mathbb{R} \cup \{+\infty\})^n$, and a subset $\mathcal{I} \subseteq [n]$, the corresponding *mixed-integer linear program* (MIP) is given by

$$\begin{aligned} \min \quad & c^\top x \\ \text{s.t.} \quad & L \leq Ax \leq U && (1.1) \\ & \ell \leq x \leq u && (1.2) \\ & x_i \in \mathbb{Z} \quad \text{for all } i \in \mathcal{I} && (1.3) \\ & x_i \in \mathbb{R} \quad \text{for all } i \in [n] \setminus \mathcal{I}. && (1.4) \end{aligned}$$

Some classes of MIP have special names in the literature:

1. Problems where $\mathcal{I} = \emptyset$ are called *linear programs* (LPs).
2. Problems where $\mathcal{I} = [n]$ are called *integer programs* (IPs).
3. IPs with $\ell_i = 0$ and $u_i = 1$ for all $i \in [n]$ are called *binary integer programs* (BIPs).

For a comprehensive treatment of LPs, we refer the reader to [9], and for a more theoretical approach, see [10]. For IPs and MIPs, we refer to [11, 12]. The art of formulating MIP models is a skill in and of itself. An engaging introduction to this topic through puzzles is given in [13].

Variables in the vector x are called *decision variables.* A point $x \in \mathbb{R}^n$ is a *feasible solution* to a MIP if it satisfies (1.1)–(1.4). We say that a MIP is *infeasible* if there exists no feasible solution. The convex hull of the set of feasible solutions is called the *integer hull* of the MIP and will be denoted throughout this thesis by $\mathcal{P}_I$. The *LP relaxation* of the MIP is the problem obtained by relaxing the integrality constraint, that is, by replacing $\mathcal{I}$ with $\emptyset$. We will denote the set of feasible solutions to the LP relaxation by $\mathcal{P}$. The optimal solution of a MIP and its LP relaxation will be

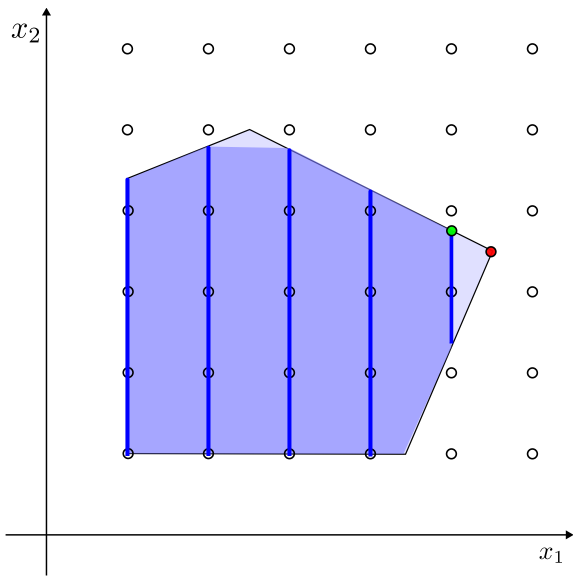

Figure 1.1: Illustration of the MIP from Example 1.2. Shown are: the feasible set of the LP relaxation (in light blue), the integer hull (in a darker shade of blue), the MIP feasible points (dark blue lines), the LP relaxation optimal solution (the red point), and an optimal solution to the MIP (the green point).

denoted by x^* and $\bar{x}$, respectively. We remark that the notation $\mathcal{P}$ is used because the feasible set of the LP relaxation is always polyhedral, and, as we will see later, the integer hull is also polyhedral in the majority of practical cases. We will discuss polyhedral sets in Section 1.4.

The sets $\mathcal{P}_I$ and $\mathcal{P}$ might not coincide, as the following example shows.

Example 1.2. Consider the IP

$$\begin{aligned} \min \quad & -x_1 \\ \text{s.t.} \quad & 7x_1 - 3x_2 \leq 28 \\ & 2x_1 + 4x_2 \leq 25 \\ & -2x_1 + 5x_2 \leq 20 \\ & x_1, x_2 \geq 1 \\ & x_1 \in \mathbb{Z} \\ & x_2 \in \mathbb{R}. \end{aligned}$$

The MIP is illustrated in Figure 1.1. The feasible set of the LP relaxation is shaded in light blue. The integer hull is shown in a darker shade of blue. The feasible solutions to the MIP are aligned along the vertical dark blue lines. The red point marks the optimal solution to the LP relaxation, while the green point indicates an optimal solution to the MIP. Note that all points that lie on the line $x_1 = 5$ are optimal.

From this example, we see that it would be desirable to tighten $\mathcal{P}$ to better approximate $\mathcal{P}_I$. To this end, we will later introduce the notion of cutting planes. A *cutting plane* is an inequality that is valid for $\mathcal{P}_I$ but not for $\mathcal{P}$; that is, it is satisfied by all points in $\mathcal{P}_I$ and violated by at least one point in $\mathcal{P}$. For most cutting planes in the literature, the violating point in $\mathcal{P}$ is an optimal solution of the LP relaxation. Cutting planes will be the focus of Chapter 2. However, due to their importance in this thesis, we introduce them already here.

Example 1.3. The inequality $x_1 \leq 5$ is a valid cutting plane for the MIP in Example 1.2. It is satisfied by all points in $\mathcal{P}_I$ and violated by the LP relaxation optimum $\bar{x} = (5.5, 3.5)$.

1.2 The Branch-and-Bound Method for Solving MIPs

In this section, we briefly outline the branch-and-bound method for solving MIPs. This method systematically divides the search space into increasingly smaller subproblems, applying the LP relaxation at each one to identify and discard inferior regions. The method was first introduced in [14] and is the method of choice for solving MIPs in modern solvers such as SCIP [1] and XPress [2].

We introduce the method by example. We modify the MIP from Example 1.2 into an IP by adding the constraint that x_2 must be integer:

$$\begin{aligned} \min \quad & -x_1 \\ \text{s.t.} \quad & 7x_1 - 3x_2 \leq 28 \\ & 2x_1 + 4x_2 \leq 25 \\ & -2x_1 + 5x_2 \leq 20 \\ & x_1, x_2 \geq 1 \\ & x_1, x_2 \in \mathbb{Z}. \end{aligned} \tag{1.5}$$

First, we solve the LP relaxation of the IP. The optimal solution to the LP relaxation is

$$\bar{x} = (5.5, 3.5).$$

Since x_2 must be an integer, any MIP-feasible solution must satisfy either $x_2 \leq 3$ or $x_2 \geq 4$. Therefore, we divide the IP into two subproblems:

$$\begin{aligned} \min \quad & -x_1 \\ \text{s.t.} \quad & 7x_1 - 3x_2 \leq 28 \\ & 2x_1 + 4x_2 \leq 25 \\ & -2x_1 + 5x_2 \leq 20 \\ & x_1, x_2 \geq 1 \\ & x_2 \leq 3 \\ & x_1, x_2 \in \mathbb{Z} \end{aligned} \quad \text{and} \quad \begin{aligned} \min \quad & -x_1 \\ \text{s.t.} \quad & 7x_1 - 3x_2 \leq 28 \\ & 2x_1 + 4x_2 \leq 25 \\ & -2x_1 + 5x_2 \leq 20. \\ & x_1, x_2 \geq 1 \\ & x_2 \geq 4 \\ & x_1, x_2 \in \mathbb{Z} \end{aligned}$$

This step is called *branching*. The feasible regions of the two subproblems are shown in Figure 1.2 as $\mathcal{P}_1$ and $\mathcal{P}_2$, respectively. Observe that $\bar{x}$ is no longer feasible for either subproblem.

We now focus on the subproblem with the additional constraint $x_2 \geq 4$. Solving its LP relaxation yields the point

$$\bar{x} \approx (5.29, 3).$$

We further branch on x_1, creating two subproblems with additional constraints $x_1 \leq 5$ and $x_1 \geq 6$, respectively. Adding the constraint $x_1 \geq 6$ leads to an infeasible LP relaxation. Hence, we say that the subproblem is *fathomable by infeasibility.*

Next, we consider the subproblem obtained by adding the constraint $x_1 \leq 5$. The optimal solution to its LP relaxation is $\bar{x} = (5, 3)$ with an objective value of -5. Since this solution is also feasible for the MIP, it is optimal for the subproblem. We say such a subproblem is *fathomable by optimality.*

We now return to the subproblem with the additional constraint $x_2 \geq 4$. Its LP relaxation has an optimal solution at $(4.5, 4)$, with an objective value of -4.5. Because we already have a feasible MIP solution with objective value -5, we conclude that there is no benefit in exploring this region: at best, we could find an integer point with an equal or greater than -4.5. Thus, we discard this subproblem. We say it is *fathomable by bounding.*

Throughout the whole branch-and-bound algorithm we construct a branch-and-bound tree. The tree constructed for this particular example is shown in Figure 1.3. We remark that the leaves of a branch-and-bound tree form a partition of the MIP feasible set: every feasible solution lies in the LP relaxation of some leaf subproblem.

A complete description of the branch-and-bound method for solving MIPs is given in Algorithm 1.

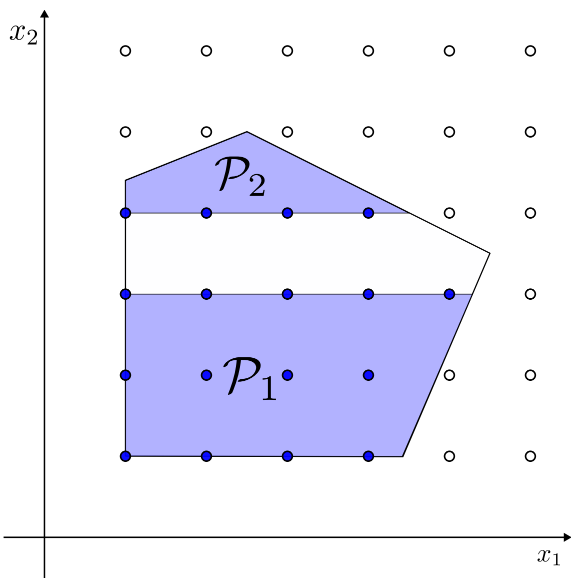

Figure 1.2: Illustration for the example in Section 1.2. Shown are the LP feasible regions of the subproblems after branching: $\mathcal{P}_1$ corresponds to the subproblem with the additional constraint $x_2 \leq 3$, and $\mathcal{P}_2$ corresponds to the subproblem with $x_2 \geq 4$.

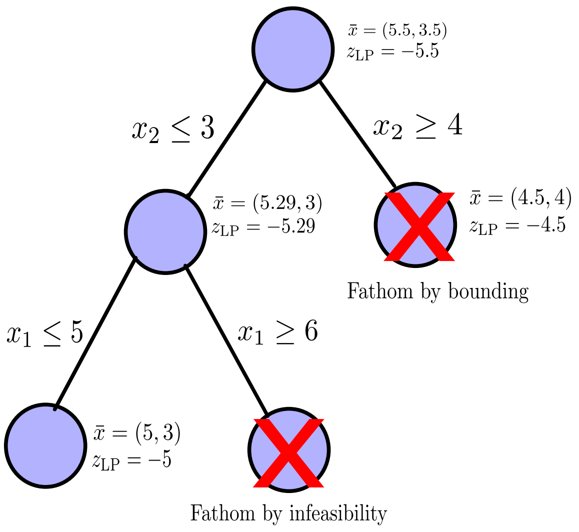

Figure 1.3: Branch-and-bound tree for the example in Section 1.2.

Algorithm 1 Branch-and-Bound method for solving MIP

Input: A bounded MIP $\mathcal{P}_0$.

Output: An optimal MIP solution x^* or or a declaration that the MIP is infeasible.

```
1. Initialise
   (a) $N \leftarrow \{\mathcal{P}_0\}$
   (b) $\underline{Z} \leftarrow +\infty$
   (c) $x^* \leftarrow$ NULL
2. While $N \neq \varnothing$
   (a) Remove a sub-problem $\mathcal{P}$ from $N$
   (b) Solve the LP relaxation of $\mathcal{P}$; let $z_{\mathrm{LP}}$ be the optimal objective value and $\bar{x}$ the optimal
       solution
         i. If the relaxation is infeasible, we can fathom by infeasibility **continue**
        ii. If $z_{\mathrm{LP}} \geq \underline{Z}$, we can fathom by bounding **continue**
   (c) If the LP solution is integer-feasible
         i. $x^* \leftarrow \bar{x}$,   $\underline{Z} \leftarrow z_{\mathrm{LP}}$
        ii. We can fathom by optimality, **continue**
   (d) Choose a fractional variable $x_j$ in the LP solution
         i. Create $\mathcal{P}_1$ with $x_j \leq \lfloor \bar{x}_j \rfloor$
        ii. Create $\mathcal{P}_2$ with $x_j \geq \lceil \bar{x}_j \rceil$
       iii. Add $\mathcal{P}_1$ and $\mathcal{P}_2$ to $N$
3. Terminate
   (a) If $x^*$ is NULL, declare the MIP infeasible
   (b) Otherwise, $x^*$ is optimal
```

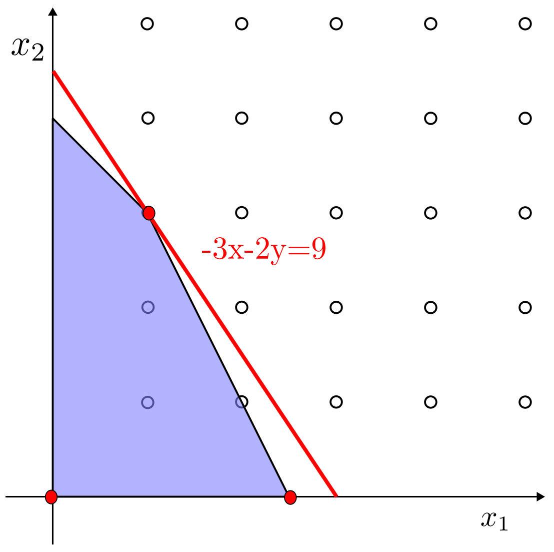

Figure 1.4: Illustration for the example from Section 1.3. Shown are the feasible region of the linear program, the level set $\{x \in \mathbb{R}^2 \mid -3x_1 - 2x_2 = -9\}$, and the points corresponding to the basic feasible solutions visited throughout the simplex algorithm (in red).

1.3 The Simplex Method for Solving Linear Programs

There are many cutting planes in the literature whose generation makes use of information obtained during the solution of the LP relaxation (see, for instance, [15], [16], [17]). The cutting planes discussed in this thesis also falls into this category. To acquaint the reader with the type of information that can be obtained from the LP solving process, we give a brief exposition of LP solving.

Most LP solvers use either the Simplex method or the Interior Point method (see [7]). In this section, we focus on the Simplex method, which is particularly useful for MIP solving because it provides access to certain structural information, known as a basis, which will be explained in the following paragraph. In contrast, the Interior Point method requires an additional routine, known as *crossover*, to recover this information.

As before, we introduce the Simplex algorithm through a numerical example. We consider the following linear program:

$$\begin{aligned} \min \quad & -3x_1 - 2x_2 \\ \text{s.t.} \quad & x_1 + x_2 \leq 4 \\ & 2x_1 + x_2 \leq 5 \\ & x_1, x_2 \geq 0. \end{aligned}$$

The feasible set of the LP is shown in Figure 1.4. The Simplex method is described for a linear program in *standard form*, that is, a linear program in the form

$$\begin{aligned} \min \quad & c^\top x \\ \text{s.t.} \quad & Ax = b \\ & x \geq 0 \end{aligned} \tag{1.6}$$

where $A \in \mathbb{R}^{m \times n}$ has full row rank, $b \in \mathbb{R}^m$, and $c \in \mathbb{R}^n$.

The conversion to standard form can be done easily by adding *slack* variables. The example linear program in standard form is

$$\begin{aligned} \min \quad & -3x_1 - 2x_2 \\ \text{s.t.} \quad & x_1 + x_2 + s_1 = 4 \\ & 2x_1 + x_2 + s_2 = 5 \\ & x_1, x_2, s_1, s_2 \geq 0. \end{aligned} \tag{1.7}$$

It is easily seen that the problem is equivalent to the original linear program, that is, they have the same objective value, and there is a one-to-one mapping between the solution sets of the two problems. The constraint matrix also has full row rank, since the columns corresponding to s_1 and s_2 are linearly independent. The goal of the Simplex algorithm is to formulate the problem into an equivalent one where an optimal solution is self-evident. In what follows, we will show that the example LP is equivalent to the LP

$$\begin{aligned} \min \quad & -9 + s_1 + s_2 \\ \text{s.t.} \quad & x_2 = 3 - 2s_1 + s_2 \\ & x_1 = 1 + s_1 - s_2 \\ & x_1, x_2, s_1, s_2 \geq 0. \end{aligned}$$

We observe that the LP attains its optimum at $(1, 3)$ with an objective value of -9 because setting either s_1 or s_2 to strictly positive values increases the objective value.

Central to the Simplex method is the notion of a *basis*.

Definition 1.4. Given a linear program in standard form, a set of decision variables $\mathcal{B}$ is called a basis if $|\mathcal{B}| = m$ and their corresponding columns in the constraint matrix are linearly independent.

Given a basis $\mathcal{B}$, decision variables that are in the basis are called *basic variables*, while those that are not are called *non-basic variables*. The set of non-basic variables is denoted by $\mathcal{N}$. Columns corresponding to basic and non-basic variables are called *basic* and *non-basic* columns, respectively. We define the *basis matrix*, denoted by B, to be the submatrix of A formed by the basic columns. We denote the submatrix of A formed by the non-basic columns by N.

By rearranging the columns of A so that those corresponding to basic variables appear before those corresponding to non-basic variables, we can rewrite the equality constraints of an LP in Form (1.6) as

$$Bx_B + Nx_N = b \tag{1.8}$$

where x_B and x_N correspond to the vectors of basic and non-basic decision variables, respectively.

The *basic solution* corresponding to basis $\mathcal{B}$ is obtained by fixing $x_N = 0$ and solving $Bx_B = b$ to obtain the values for all basic variables. The basic solution is a *basic feasible solution* if, additionally, $x_B \geq 0$—that is, it is a feasible solution to our LP. In this case, we say the basis is a *feasible* basis.

In our example LP, we observe that $\mathcal{B}_1 := \{s_1, s_2\}$ and $\mathcal{B}_2 := \{x_1, s_2\}$ are two bases for the LP. A straightforward computation shows that their corresponding basic solutions (x_1, x_2, s_1, s_2) are $(0, 0, 4, 5)$ and $(4, 0, 0, -3)$, respectively. Hence, $\mathcal{B}_1$ is a feasible basis, while $\mathcal{B}_2$ is not. Note that the points corresponding to basic solutions in the original LP coincide with vertices of the feasible region.

Another way of writing the equality constraints of an LP in Form (1.6) is to use the fact that B is an invertible matrix and multiply both sides of (1.8) by B^{-1} to obtain

$$x_B = -B^{-1}Nx_N + B^{-1}b, \tag{1.9}$$

which expresses each basic variable as a linear combination of the non-basic variables plus some constant.

We can now rewrite $c^\top x$ as follows:

$$\begin{aligned} c^\top x &= c_B^\top x_B + c_N^\top x_N \\ &= -c_B^\top B^{-1}Nx_N + c_B^\top B^{-1}b + c_N^\top x_N \\ &= c_B^\top B^{-1}b + (c_N^\top - c_B^\top B^{-1}N)x_N, \end{aligned}$$

thus expressing the cost function in terms of only the non-basic variables. We define the dual vector y by $y := c_B^\top B^{-1}$. The coefficient of a decision variable x_i in this rewritten form is called the reduced cost of x_i, denoted by $\bar{c}_i$. More precisely, the reduced cost is given by

$$\bar{c}_i := \begin{cases} 0 & \text{if } x_i \in \mathcal{B}, \\ c_i - y^\top A_i & \text{if } x_i \notin \mathcal{B}, \end{cases}$$

where A_i is the ith column of A. The LP in *tableau form* with respect to basis $\mathcal{B}$ is given by

$$\begin{aligned} \min \quad & c_B^\top B^{-1} b + \bar{c}_N^\top x_N \\ \text{s.t.} \quad & x_B = -B^{-1} N x_N + B^{-1} b \\ & x \geq 0. \end{aligned} \tag{1.10}$$

At the beginning of the Simplex method, we are given a starting feasible basis. We use $\mathcal{B}_1 := \{s_1, s_2\}$ as an example here, whose corresponding basic solution is $(0,0,4,5)$. The tableau form of our LP with respect to $\mathcal{B}_1$ can be obtained by a simple rearrangement of the equality constraints in (1.7). We obtain

$$\begin{aligned} \min \quad & -3x_1 - 2x_2 && (1.11) \\ \text{s.t.} \quad & s_1 = 4 - x_1 - x_2 && (1.12) \\ & s_2 = 5 - 2x_1 - x_2 && (1.13) \\ & x_1, x_2, s_1, s_2 \geq 0. \end{aligned}$$

The Simplex method is an iterative algorithm in which we strategically iterate over the set of possible bases through a so-called *pivot* operation. During a pivot, we obtain a new basis by exchanging a non-basic variable with a basic variable. The operation must ensure that the new basis is also feasible. At the basis $\mathcal{B}_1$, we can bring the column associated with x_1 or x_2 into the basis. Whichever variable enters is allowed to take on a nonzero value. Increasing either x_1 or x_2 decreases our objective value because the reduced costs of both x_1 and x_2 are negative. Note that decreasing x_1 or x_2 leads to infeasibility. For our example, we bring x_1 into the basis.

From (1.12), we see that if we increase x_1 by 1, we must decrease s_1 by 1 to maintain equality. To remain feasible, we can increase x_1 by at most 4, beyond which s_1 becomes negative. Applying the same reasoning to (1.13), we find that we can increase x_1 by at most 2.5 before s_2 becomes negative. Thus, s_2 is *blocking* and should leave the basis.

We now update the tableau for the new basis. First, we rewrite the equality constraint for the leaving variable in terms of the entering variable, i.e., we rearrange (1.13) into

$$x_1 = 2.5 - 0.5s_2 - 0.5x_2. \tag{1.14}$$

Next, we substitute this into the other constraints and the objective function. Substituting (1.14) into (1.12) yields

$$s_1 = 1.5 + 0.5s_2 - 0.5x_2,$$

and substituting into (1.11) yields

$$-3x_1 - 2x_2 = -7.5 + 1.5s_2 - 0.5x_2.$$

Combining all three, we rewrite the LP as

$$\begin{aligned} \min \quad & -7.5 + 1.5s_2 - 0.5x_2 \\ \text{s.t.} \quad & s_1 = 1.5 + 0.5s_2 - 0.5x_2 \\ & x_1 = 2.5 - 0.5s_2 - 0.5x_2 \\ & x_1, x_2, s_1, s_2 \geq 0. \end{aligned}$$

Clearly, this LP is equivalent to the original LP. The new basic feasible solution is $(2.5, 0, 1.5, 0)$ with an objective value of -7.5. The reduced cost is $(0, -0.5, 0, 1.5)$. This completes one simplex iteration.

The second iteration is similar, though selection of the entering variable requires more care. Our candidates are s_2 and x_2. Bringing s_2 into the basis would increase the objective value, as its reduced cost is positive and feasibility requires $s_2 \geq 0$. Thus, we discard this option. In contrast, x_2 may decrease the objective, so we select x_2 as the entering variable. As in the previous iteration, we find that s_1 is blocking and must leave the basis. Performing the pivot yields the LP

$$\begin{aligned} \min \quad & -9 + s_1 + s_2 \\ \text{s.t.} \quad & x_2 = 3 - 2s_1 + s_2 \\ & x_1 = 1 + s_1 - s_2 \\ & x_1, x_2, s_1, s_2 \geq 0. \end{aligned}$$

Since both reduced costs are positive, increasing either s_1 or s_2 will increase the objective. Thus, the current solution $(1, 3, 0, 0)$ is optimal. Examining the sequence of basic solutions visited throughout the algorithm, we see in Figure 1.4 that the Simplex method can be viewed as traversing from one vertex to another of the feasible region.

We now discuss two additional edge cases that may arise during a pivot operation. First, suppose our optimization problem is

$$\begin{aligned} \min \quad & -7.5 + 1.5s_2 - 0.5x_2 \\ \text{s.t.} \quad & s_1 = 1.5 + 0.5s_2 + 0.5x_2 \\ & x_1 = 2.5 - 0.5s_2 + 0.5x_2 \\ & x_1, x_2, s_1, s_2 \geq 0. \end{aligned}$$

and we want to bring x_2 into the basis. Since the coefficient of x_2 is positive in both equality constraints, we can increase x_2 indefinitely without s_1 or x_1 ever going negative. In this case, we can declare the LP to be *unbounded*, since our objective value improves as we increase x_2.

Second, suppose our optimization problem is

$$\begin{aligned} \min \quad & -7.5 + 1.5s_2 - 0.5x_2 \\ \text{s.t.} \quad & s_1 = 0 + 0.5s_2 - 0.5x_2 \\ & x_1 = 2.5 - 0.5s_2 + 0.5x_2 \\ & x_1, x_2, s_1, s_2 \geq 0. \end{aligned}$$

and we want to bring x_2 into the basis. Then, we cannot increase x_2, since s_1 is blocking with a value of 0. In this case, we call the basic feasible solution *degenerate*. There are special rules for handling degenerate basic feasible solutions. We refer to [9] for a more comprehensive treatment. It suffices to know that in this case, we can still make progress by performing a pivot and rewriting the system of equalities using another set of basic variables.

We formalize the Simplex algorithm in Algorithm 2. We also include the standard terminology used for each step in the literature (see [18] and [19]).

The topics of how the updating in Step 2 is performed in a numerically stable way, the selection of the entering variable in Step 3, and the tie-breaking rule for Step 4 have all been researched extensively. For a more in-depth study, we refer to [7, 18, 19, 20, 21].

1.4 Dual Representation of Polyhedra

A central result used throughout this thesis is the dual representation theorem for pointed polyhedra. We review the necessary concepts here. For a more in-depth study, we refer to [12], [22, Chapters 1–2], and [23]. Proofs of the theorems presented here can be found in [12].

We first introduce *half-spaces* and *hyperplanes*.

Definition 1.5. A set $\mathcal{H} \subseteq \mathbb{R}^n$ is a *closed half-space* if there exists a vector $\alpha \in \mathbb{R}^n$ and a scalar $\beta \in \mathbb{R}$ such that

$$\mathcal{H} = \{x \in \mathbb{R}^n \mid \alpha^\top x \leq \beta\}.$$

We call $\mathcal{H}$ a hyperplane if the inequality is replaced by an equality.

Algorithm 2 Simplex Method for a Minimisation LP

Input: constraint matrix $A \in \mathbb{R}^{m \times n}$, right–hand side $b \in \mathbb{R}^m$, cost vector $c \in \mathbb{R}^n$, and an initial feasible basis $\mathcal{B}$.
Output: an optimal solution x^* or a declaration that the LP is unbounded.

```
1. Convert the LP into tableau form with respect to the initial basis $\mathcal{B}$.
2. Update Tableau
    (a) Update the tableau so that it reflects the current basis.
3. Pricing
    (a) If all reduced costs are non-negative, go to Step 6.
    (b) Otherwise select a non-basic variable with a negative reduced cost to enter the basis.
4. Ratio test
    (a) Identify the blocking variable.
    (b) If no blocking variable exists, declare the LP unbounded and stop.
    (c) Otherwise the blocking variable leaves the basis.
5. Basis Update
    (a) Update the basis and return to Step 2.
6. Terminate
    (a) The current basic feasible solution is optimal.
```

To link back to our discussion on LPs, half-spaces naturally arise from the set of points satisfying a single inequality constraint, while hyperplanes arise from the set of points satisfying a single equality constraint.

Definition 1.6. A set $\mathcal{P}$ is an *$\mathcal{H}$-Polyhedron* if it can be written as the intersection of finitely many closed half-spaces, that is, if there exist $m \in \mathbb{N}$, a matrix $A \in \mathbb{R}^{m\times n}$, and a vector $b \in \mathbb{R}^m$ such that

$$\mathcal{P} = \{x \in \mathbb{R}^n \mid Ax \leq b\}.$$

It is easy to see that the feasible set of the LP relaxation of a MIP is an $\mathcal{H}$-Polyhedron.

Definition 1.7. Let $\mathcal{P}$ be an $\mathcal{H}$-polyhedron. A set $\mathcal{F}$ is a *face* of $\mathcal{P}$ if there exists a hyperplane $\mathcal{H}$ such that

$$\mathcal{F} = \mathcal{P} \cap \mathcal{H}.$$

By this definition, $\mathcal{P}$ and $\emptyset$ are also faces. Faces of $\mathcal{P}$ that are neither $\emptyset$ nor $\mathcal{P}$ itself are called *proper* faces. The inclusion-wise maximal proper faces of $\mathcal{P}$ are called *facets*, while faces of $\mathcal{P}$ consisting of a single point are called *vertices*. We remark here that the set of basic feasible solutions introduced earlier is exactly the set of vertices of a polyhedron. A polyhedron is *pointed* if it has at least one vertex. We illustrate these concepts in Figure 1.5.

Definition 1.8. Let $\mathcal{V} := \{v_1, \ldots, v_k\} \subseteq \mathbb{R}^n$. We define the *convex hull* of $\mathcal{V}$ as the set

$$\operatorname{conv}(\mathcal{V}) = \left\{ \sum_{i=1}^{k} \lambda_i v_i \;\middle|\; \lambda_1, \ldots, \lambda_k \geq 0 \text{ and } \sum_{i=1}^{k} \lambda_i = 1 \right\}.$$

We define the *conic hull* of $\mathcal{V}$ as the set

$$\operatorname{cone}(\mathcal{V}) = \left\{ \sum_{i=1}^{k} \lambda_i v_i \;\middle|\; \lambda_1, \ldots, \lambda_k \geq 0 \right\}.$$

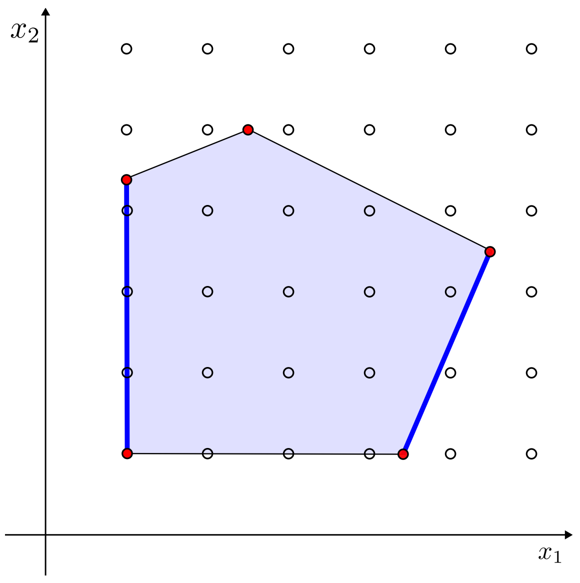

Figure 1.5: The feasible region of the LP relaxation of the MIP from Example 1.2 is a polyhedron. It has five vertices and five facets. All five vertices are shown as red dots, while two of the five facets are marked with blue lines.

We observe that the polyhedron in Figure 1.5 coincides with the convex hull of its vertices. We will see shortly that this is always the case for bounded pointed polyhedra. We now give an alternative definition of polyhedra. We will then see that the two definitions are closely related.

Definition 1.9. A set $\mathcal{P} \subseteq \mathbb{R}^n$ is a $\mathcal{V}$-*Polyhedron* if there exist finite sets $\mathcal{V}, \mathcal{R} \subseteq \mathbb{R}^n$ such that

$$\mathcal{P} = \operatorname{conv}(\mathcal{V}) + \operatorname{cone}(\mathcal{R}),$$

where $+$ refers to the Minkowski sum of the two sets.

The two definitions are equivalent if the $\mathcal{H}$-polyhedron is pointed.

Theorem 1.10. Let $\mathcal{P} \subseteq \mathbb{R}^n$. Then $\mathcal{P}$ is a $\mathcal{V}$-Polyhedron if and only if it is a pointed $\mathcal{H}$-Polyhedron.

Since we will be working with pointed polyhedra throughout this thesis, we may switch between the two representations given by the definitions of $\mathcal{H}$- and $\mathcal{V}$-polyhedra. We will also say that a set $\mathcal{P}$ is polyhedral when a specific representation is not needed.

If $\mathcal{P}$ is a pointed $\mathcal{H}$-Polyhedron, then we can give a pair of finite sets $\mathcal{V}$ and $\mathcal{R}$ which satisfy the criterion described in Definition 1.9.

Definition 1.11. Let $\mathcal{P}$ be a polyhedron. The *recession cone* of $\mathcal{P}$ is the set

$$\operatorname{rec}(\mathcal{P}) = \{d \in \mathbb{R}^n \mid \exists x_0 \in \mathcal{P} : \forall \lambda \geq 0,\ x_0 + \lambda d \in \mathcal{P}\}.$$

A vector $d \in \operatorname{rec}(\mathcal{P})$ is an *extreme ray* of $\mathcal{P}$ if there exist no $\lambda, \mu > 0$ and linearly independent vectors $u, v \in \operatorname{rec}(\mathcal{P})$ such that

$$d = \lambda u + \mu v.$$

The next theorem states that a polyhedron has finitely many vertices and that there exists a finite set of extreme rays that generate the recession cone.

Theorem 1.12. Let $\mathcal{P}$ be a polyhedron. The set of vertices of $\mathcal{P}$ is finite. Furthermore, there exists a finite set of extreme rays $\mathcal{R}$ such that $\operatorname{rec}(\mathcal{P}) = \operatorname{cone}(\mathcal{R})$.

We now state the dual representation theorem for polyhedra.

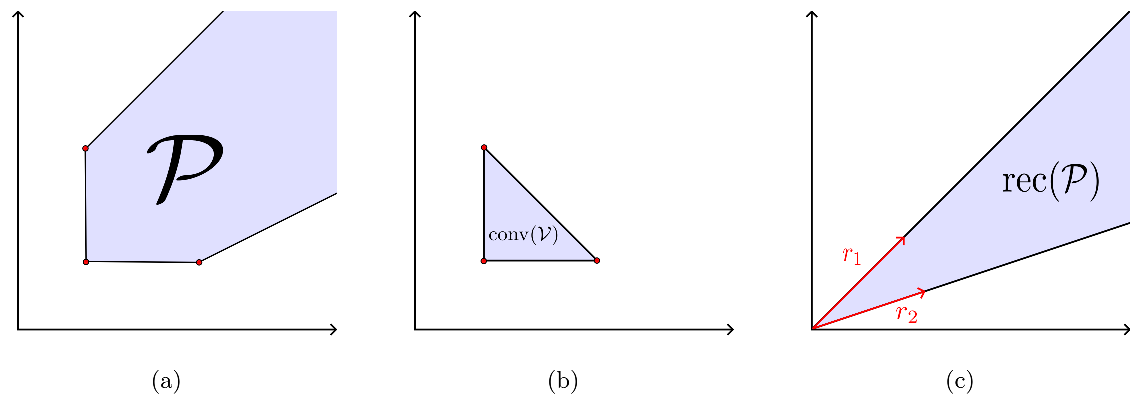

Figure 1.6: Visualization for Theorem 1.13.

Theorem 1.13. Let $\mathcal{P}$ be a pointed $\mathcal{H}$-Polyhedron. Furthermore, let $\mathcal{V}$ be the set of vertices of $\mathcal{P}$ and $\mathcal{R}$ a finite set of generators of $\mathrm{rec}(\mathcal{P})$. Then,

$$\mathcal{P} = \mathrm{conv}(\mathcal{V}) + \mathrm{rec}(\mathcal{P}) = \mathrm{conv}(\mathcal{V}) + \mathrm{cone}(\mathcal{R}).$$

Figure 1.6 illustrates Theorem 1.13. The vertices of the polyhedron $\mathcal{P}$ are marked with red dots in Panels (a) and (b). Panel (b) shows the convex hull of these vertices, while Panel (c) illustrates the recession cone of $\mathcal{P}$. The two rays r_1 and r_2, drawn in red, generate the cone $\mathrm{rec}(\mathcal{P})$.

1.5 The Corner Polyhedron

We consider the feasible set of the LP relaxation of a MIP. In this section, we introduce a further relaxation of this set. The motivation is that, for a given MIP, the number of vertices and rays of the feasible region of the LP relaxation may be exponential in the number of inequalities defining the MIP. We aim for a relaxation of this set whose $\mathcal{V}$-polyhedral representation consists of only one vertex and as many rays as the problem dimension. We assume that the LP relaxation has been solved using the Simplex method.

At the end of the Simplex method, we are given an optimal basis. Let n be the number of variables in our problem and m be the number of rows (which is the same as the number of basic variables). We rearrange the decision variables such that the first m variables $x_1, \dots, x_m$ are basic. Then, our optimal tableau has the form:

$$\begin{aligned} x_1 &= a_{1,m+1}x_{m+1} + a_{1,m+2}x_{m+2} + \cdots + a_{1,n}x_n \\ x_2 &= a_{2,m+1}x_{m+1} + a_{2,m+2}x_{m+2} + \cdots + a_{2,n}x_n \\ &\vdots \\ x_m &= a_{m,m+1}x_{m+1} + a_{m,m+2}x_{m+2} + \cdots + a_{m,n}x_n \end{aligned} \tag{1.15}$$

where $a_{i,j}$ are the tableau entries. Note that the feasible set of our LP relaxation is exactly the set of points satisfying the system of equalities (1.15) together with $x \geq 0$.

We define the corner polyhedron relaxation as the relaxation obtained by removing the constraints $x_B \geq 0$, that is, the corner polyhedron is defined by the system of equalities (1.15) along with $x_N \geq 0$.

We illustrate this definition using the example from Section 1.3. Recall that at the end of the Simplex algorithm, our optimal tableau is

$$\begin{aligned} x_2 &= 3 - 2s_1 + s_2 \\ x_1 &= 1 + s_1 - s_2. \end{aligned} \tag{1.16}$$

Together with the constraints $x_1, x_2, s_1, s_2 \geq 0$, we obtain the feasible set of our example LP, illustrated in Figure 1.4 (after projection to x_1 and x_2). We now relax the non-negativity requirements

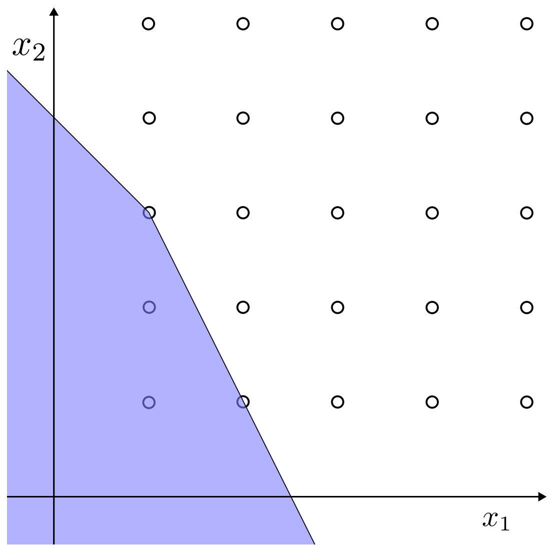

Figure 1.7: The corner polyhedron relaxation at the optimal basis from the example in Section 1.3, projected onto the x_1–x_2 space.

on the basic variables, that is, x_1 and x_2. The relaxed set, projected to x_1 and x_2, is shown in Figure 1.7.

The $\mathcal{V}$-polyhedral representation of the corner polyhedron is also simple to describe. The unique vertex is the basic feasible solution associated with the optimal basis. The polyhedron has $n - m$ extreme rays, each corresponding to moving one of the non-basic variables away from zero. For $i \in [n - m]$, the i-th extreme ray is given by

$$r_k^i := \begin{cases} a_{k,m+i}, & \text{if } k = 1, \ldots, m \\ 1, & \text{if } k = m + i \\ 0, & \text{otherwise.} \end{cases}$$

In our example, the vertex of the corner polyhedron relaxation is $(1, 3, 0, 0)$. The first ray corresponds to increasing s_1. To increase s_1 by 1, we need to decrease x_2 by 2 and increase x_1 by 1 to maintain the equalities in (1.16). Hence, the first extreme ray is $(1, -2, 1, 0)$. Similarly, we obtain the second extreme ray to be $(-1, 1, 0, 1)$.

Chapter 2

Foundations of $\mathcal{V}$-Polyhedral Disjunctive Cuts

In this chapter, we provide the theoretical background on $\mathcal{V}$-polyhedral disjunctive cuts (VPCs). In Section 2.1, we provide a brief overview of cutting planes and their role in the solution of mixed-integer linear programs (MIPs). VPCs are a subset of the broader class of cutting planes known as disjunctive cutting planes, which are discussed in Section 2.2. The theory of VPCs is developed from Section 2.3 to 2.5.

2.1 Cutting Planes in Mixed-Integer Programming

As in the previous chapter, we consider a MIP of the form:

$$\begin{aligned} \min \quad & c^\top x \\ \text{s.t.} \quad & L \leq Ax \leq U \\ & \ell \leq x \leq u \\ & x_i \in \mathbb{Z} \quad \text{for all } i \in \mathcal{I} \\ & x_i \in \mathbb{R} \quad \text{for all } i \in [n] \setminus \mathcal{I} \end{aligned} \tag{2.1}$$

where $c \in \mathbb{R}^n$, $A \in \mathbb{R}^{m \times n}$, $L \in (\mathbb{R} \cup \{-\infty\})^m$, $U \in (\mathbb{R} \cup \{+\infty\})^m$, $\ell \in (\mathbb{R} \cup \{-\infty\})^n$, $u \in (\mathbb{R} \cup \{+\infty\})^n$, and $\mathcal{I} \subseteq [n]$. The integer hull of the MIP is denoted by $\mathcal{P}_I$, and the feasible set of the LP relaxation by $\mathcal{P}$. In general, $\mathcal{P}_I \subseteq \mathcal{P}$, but $\mathcal{P} \neq \mathcal{P}_I$.

We introduce valid inequalities for polyhedra.

Definition 2.1. Given a polyhedron $\mathcal{P}$, an inequality $\alpha^\top x \leq \beta$ is *valid* for $\mathcal{P}$ if it is satisfied by all $x \in \mathcal{P}$.

We now define cutting planes for MIPs.

Definition 2.2. Given a MIP, an inequality $\alpha^\top x \leq \beta$ is *valid* for the MIP if it is valid for $\mathcal{P}_I$. It is a *cutting plane* if, in addition, it is violated by at least one point in $\mathcal{P} \setminus \mathcal{P}_I$.

Typically, the point that violates a cutting plane is the optimal solution to the LP relaxation. For the rest of the thesis, we use the terms cutting planes and cuts interchangeably. By adding cutting planes to a MIP, we tighten $\mathcal{P}$ to better approximate $\mathcal{P}_I$. If, after adding finitely many cutting planes, the relaxation satisfies $\mathcal{P} = \mathcal{P}_I$, then we say that we have obtained a *perfect formulation* of the MIP. In this case, solving the LP relaxation yields a solution that is both feasible and optimal for the MIP.

Typical examples of MIPs for which a perfect formulation is known are those arising in matching problems [24]. However, in general, a perfect formulation for a given MIP is unknown. In fact, $\mathcal{P}_I$

may not even be a polyhedron, in which case a perfect formulation cannot exist. The following theorem provides conditions under which $\mathcal{P}_I$ is a polyhedron.

Theorem 2.3 (Meyer [25])**.** Given a MIP of the form (2.1), if at least one of the following holds:

(i) the entries of A, L, U, ℓ, and u are rational,

(ii) $\mathcal{P}$ is a polytope,

then $\mathcal{P}_I$ is a polyhedron.

There exist theoretical methods for solving a MIP by adding cutting planes that guarantee finite termination. For integer programs with an integral cost vector, this can be done through the use of *Gomory cuts* [26]. The method described in [26] is recursive: it iteratively generates a new cut using the original problem and previously generated cuts. An extension of the method allows a relaxed condition in which the objective depends only on the integer-constrained variables. In practice, however, generating cuts through such a recursive procedure introduces numerical issues and leads to a weakening of cut strength in later iterations [27, 16]. This limitation is significant, as it motivates the development of strong cutting planes that can be derived directly from the original problem formulation, such as the approach proposed in this thesis.

For this reason, most solvers use cutting planes primarily to reduce the search space within the branch-and-bound algorithm, rather than to find a perfect formulation. The resulting combination is usually called *branch-and-cut*. Computational results in the literature show that adding cutting planes can significantly improve the performance of MIP solvers; see, for example, [3]. In what follows, we describe a method for generating cutting planes for general MIPs. There are also application-specific cutting planes in the literature that exploit the structure of certain MIPs. For a general overview of different cutting plane generation techniques, we refer to [28] and [29].

As a final remark, the selection of cutting planes to add to the MIP formulation is an important and active area of research. Adding too many cuts may slow down the solution time for the LP relaxation. For an overview of cut selection strategies, we refer to [30], [31] and [32].

2.2 Disjunctive Cutting Planes

To aid intuition, we begin this section with the following example.

Example 2.4. We consider the IP

$$\begin{aligned} \max \quad & x_1 \\ \text{s.t.} \quad & 7x_1 - 3x_2 \le 28 \\ & 2x_1 + 4x_2 \le 25 \\ & -2x_1 + 5x_2 \le 20 \\ & x_1, x_2 \ge 1 \\ & x_1, x_2 \in \mathbb{Z}. \end{aligned} \tag{2.2}$$

The feasible set of the LP relaxation, the integer hull, and the LP feasible points are illustrated in Figure 2.1. We can disregard the objective for now and concern ourselves with finding cutting planes for the IP. Using $\mathcal{P}$ to denote the feasible set of the LP relaxation, observe that all integer feasible points lie in either $\mathcal{D}_1 := \mathcal{P} \cap \{x \in \mathbb{R}^2 \mid x_2 \le 3\}$ or $\mathcal{D}_2 := \mathcal{P} \cap \{x \in \mathbb{R}^2 \mid x_2 \ge 4\}$. The sets $\mathcal{D}_1$ and $\mathcal{D}_2$ are illustrated in panel (b). Observe that both $\mathcal{D}_1$ and $\mathcal{D}_2$ are polyhedral sets.

Definition 2.5. A set is a *disjunctive set* if it can be written as a union of polyhedra. The individual polyhedra are called *disjunctive terms.* The closure of the convex hull of a disjunctive set is called the *disjunctive hull.*

We note that the terms *disjunctive set* and *disjunction* are used interchangeably in the literature. We follow this convention in this thesis. For a given disjunctive set $\mathcal{D}$, we will denote the disjunctive hull by $\bar{\mathcal{D}}$. We now continue the discussion from Example 2.4.

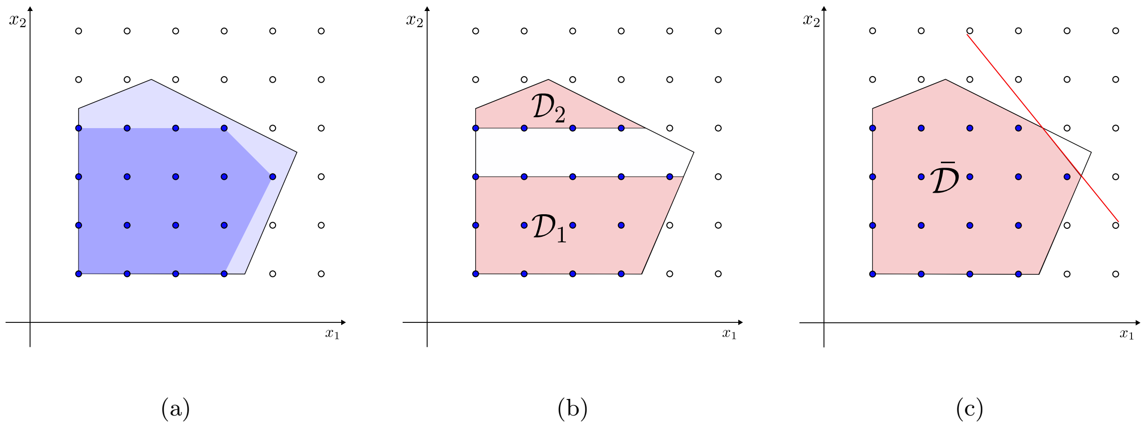

Figure 2.1: Figure for Example 2.4 shows in panel (a) the feasible region of the MIP, shaded in light blue, with the integer hull in darker blue and integer-feasible points marked as filled blue dots, in panel (b) the sets $\mathcal{D}_1$ and $\mathcal{D}_2$, and in panel (c) the disjunctive hull of the disjunctive set $\mathcal{D} := \mathcal{D}_1 \cup \mathcal{D}_2$ and a cut satisfied by all points in $\bar{\mathcal{D}}$.

Example 2.6. We have that $\mathcal{D} := \mathcal{D}_1 \cup \mathcal{D}_2$ is a disjunctive set. $\bar{\mathcal{D}}$ is shown in panel (c) of Figure 2.1.

In Section 2.3, we will show that the disjunctive hull of a disjunctive set is a polyhedron. We now define *valid* disjunctive sets.

Definition 2.7. Given a MIP, a disjunctive set $\mathcal{D}$ is *valid* for the MIP if $\mathcal{P}_I \subseteq \bar{\mathcal{D}}$.

The following lemma directly follows from the definition of a valid disjunctive set.

Lemma 2.8. Given a MIP and a valid disjunctive set $\mathcal{D}$, if an inequality is satisfied by all points $x \in \bar{\mathcal{D}}$, then the inequality is valid for the MIP. If, additionally, an element of $\mathcal{P}$ violates the inequality, then the inequality is a cut.

The foundational work on disjunctive sets is due to Balas [33]. Cuts derived from valid inequalities for disjunctive hulls, as described in Lemma 2.8, are commonly known as *disjunctive cutting planes*. We now return to Example 2.4.

Example 2.9. The inequality $14x_1 + 11x_2 \leq 107$ is satisfied by all points in $\bar{\mathcal{D}}$. Hence, it is a valid inequality for the MIP. The inequality is illustrated in panel (c) of Figure 2.1. Furthermore, the LP optimum $(5.5, 3.5)$ is violated by the inequality. Hence, the inequality is a cut.

The disjunctive set used in Example 2.4 is commonly referred to in the literature as a *split disjunction*. A cut that is valid for the disjunctive hull of a split disjunction is often called a *split cut*. Split cuts are of particular theoretical interest because the *Gomory mixed-integer cut* (GMIC) can be derived by strengthening split cuts [34].

From this point on, we assume that a MIP and a valid disjunctive set are given and focus on generating valid inequalities for the disjunctive hull.

2.3 $\mathcal{V}$-Polyhedral Disjunctive Cuts

$\mathcal{V}$-polyhedral cuts were first introduced in [4]. The key idea behind $\mathcal{V}$-polyhedral disjunctive cuts is to represent the disjunctive hull using the $\mathcal{V}$-polyhedral representation. It turns out that the $\mathcal{V}$-polyhedral representation can be easily constructed from the $\mathcal{V}$-polyhedral representations of each individual disjunctive term. The following proof is a modified version of the one presented in [12].

Theorem 2.10. Given a disjunctive set $\mathcal{D} := \bigcup_{i=1}^{N} \{x \in \mathbb{R}^n \mid D_i x \leq d_i\}$ where each disjunctive term is pointed, non-empty, and admits a $\mathcal{V}$-polyhedral representation

$$\{x \in \mathbb{R}^n \mid D_i x \leq d_i\} = \operatorname{conv}(\mathcal{V}_i) + \operatorname{cone}(\mathcal{R}_i).$$

Then

$$\bar{\mathcal{D}} = \operatorname{conv}\left(\bigcup_{i=1}^{N} \mathcal{V}_i\right) + \operatorname{cone}\left(\bigcup_{i=1}^{N} \mathcal{R}_i\right).$$

In particular, $\bar{\mathcal{D}}$ is a polyhedron admitting a $\mathcal{V}$-polyhedral representation.

Proof. Let

$$T := \operatorname{conv}\left(\bigcup_{i=1}^{N} \mathcal{V}_i\right) + \operatorname{cone}\left(\bigcup_{i=1}^{N} \mathcal{R}_i\right).$$

We first prove that $\bar{\mathcal{D}} \subseteq T$. Since T is a $\mathcal{V}$-polyhedron, it is closed and convex. Observe that, for each $i \in [N]$, we have

$$\{x \in \mathbb{R}^n \mid D_i x \le d_i\} \subseteq T,$$

which implies

$$\bigcup_{i=1}^{N} \{x \in \mathbb{R}^n \mid D_i x \le d_i\} \subseteq T.$$

Since $\bar{\mathcal{D}}$ is the closure of the convex hull of $\mathcal{D}$, it follows immediately that $\bar{\mathcal{D}} \subseteq T$. It remains to prove that $T \subseteq \bar{\mathcal{D}}$.

For each $i \in [N]$, we denote the elements of $\mathcal{V}_i$ by $v_1^i, \ldots, v_{m_i}^i$ and the elements of $\mathcal{R}_i$ by $r_1^i, \ldots, r_{n_i}^i$. Let $x^* \in T$. We have

$$x^* = \sum_{i=1}^{N} \sum_{j=1}^{m_i} \alpha_{ij} v_j^i + \sum_{i=1}^{N} \sum_{j=1}^{n_i} \beta_{ij} r_j^i$$

with $\sum_{i=1}^{N} \sum_{j=1}^{m_i} \alpha_{ij} = 1$, $\alpha \ge 0$, $\beta \ge 0$.

For $i \in [N]$, we define $z_i := \sum_{j=1}^{m_i} \alpha_{ij}$ and $\mathcal{I} := \{i \in [N] \mid z_i > 0\}$. Furthermore, for all $i \in \mathcal{I}$, we define

$$y_i := \sum_{j=1}^{m_i} \frac{1}{z_i} \alpha_{ij} v_j^i + \sum_{j=1}^{n_i} \frac{1}{z_i} \beta_{ij} r_j^i.$$

Observe that $y_i \in \{x \in \mathbb{R}^n \mid D_i x \le d_i\}$ for all $i \in \mathcal{I}$. We can now rewrite

$$x^* = \sum_{i \in \mathcal{I}} z_i y_i + \sum_{i \in [N] \setminus \mathcal{I}} \sum_{j=1}^{n_i} \beta_{ij} r_j^i.$$

For all $\epsilon > 0$ small enough so that $z_i - \frac{1}{|\mathcal{I}|}\epsilon > 0$ for all $i \in \mathcal{I}$, we define

$$x_\epsilon := \sum_{i \in \mathcal{I}} \left(z_i - \frac{1}{|\mathcal{I}|}\epsilon\right) y_i + \sum_{i \in [N] \setminus \mathcal{I}} \frac{\epsilon}{N - |\mathcal{I}|} \left(v_1^i + \frac{N - |\mathcal{I}|}{\epsilon} \sum_{j=1}^{n_i} \beta_{ij} r_j^i\right).$$

Note that, if $\mathcal{I} = [N]$, the second sum is empty. Since, for all $i \in [N] \setminus \mathcal{I}$, it holds that

$$v_1^i + \frac{N - |\mathcal{I}|}{\epsilon} \sum_{j=1}^{n_i} \beta_{ij} r_j^i \in \{x \in \mathbb{R}^n \mid D_i x \le d_i\}$$

and

$$\sum_{i \in \mathcal{I}} \left(z_i - \frac{1}{|\mathcal{I}|}\epsilon\right) + \sum_{i \in [N] \setminus \mathcal{I}} \frac{\epsilon}{N - |\mathcal{I}|} = 1,$$

we have

$$x_\epsilon \in \operatorname{conv}(\mathcal{D}).$$

Taking the limit as $\epsilon \to 0$, we have $x^* \in \bar{\mathcal{D}}$. □

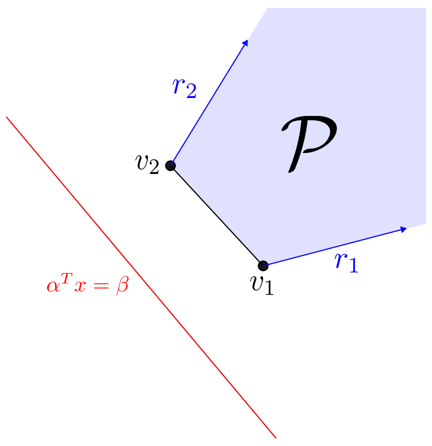

Figure 2.2: Visualization of Theorem 2.12.

The following example shows that taking the closure is necessary, as in general, the convex hull of the disjunctive set is not closed.

Example 2.11. Let $\mathcal{D}_1 = \{(x, 0) \mid x \geq 0\}$ and $\mathcal{D}_2 = \{(1, y) \mid y \geq 1\}$. The convex hull of the disjunctive set $\mathcal{D}_1 \cup \mathcal{D}_2$ is the set $\{(x, y) \mid x > 0, y \geq 0\}$, which is not closed.

The following theorem gives us a method for finding valid inequalities for a general polyhedron that admits a $\mathcal{V}$-polyhedral representation. We expand upon the proof in [4].

Theorem 2.12. Let $\mathcal{P} = \text{conv}(\mathcal{V}) + \text{cone}(\mathcal{R}) \subseteq \mathbb{R}^n$ be a pointed polyhedron admitting a $\mathcal{V}$-polyhedral representation. An inequality $\alpha^\top x \geq \beta$ is valid for $\mathcal{P}$ if and only if it satisfies:

(i) $\alpha^\top v \geq \beta$ for all $v \in \mathcal{V}$

(ii) $\alpha^\top r \geq 0$ for all $r \in \mathcal{R}$

Proof. Let $\alpha^\top x \geq \beta$ be an inequality satisfying the conditions, and let $x \in \mathcal{P}$ be arbitrary. We have

$$x = \lambda_1 v_1 + \cdots + \lambda_k v_k + \mu_1 r_1 + \cdots + \mu_\ell r_\ell$$

where $k, \ell \in \mathbb{N}$, $v_1, \ldots, v_k \in \mathcal{V}$, $r_1, \ldots, r_\ell \in \mathcal{R}$, $\lambda, \mu \geq 0$, and $\sum_{i=1}^k \lambda_i = 1$. Then,

$$\begin{aligned}\alpha^\top x &= \alpha^\top \left(\sum_{i=1}^k \lambda_i v_i + \sum_{i=1}^\ell \mu_i r_i \right) \\ &= \sum_{i=1}^k \lambda_i \alpha^\top v_i + \sum_{i=1}^\ell \mu_i \alpha^\top r_i \\ &\geq \sum_{i=1}^k \lambda_i \beta \\ &= \beta\end{aligned}$$

Conversely, let $\alpha^\top x \geq \beta$ be a valid inequality for $\mathcal{P}$. Then, since $\mathcal{V} \subseteq \mathcal{P}$, we have $\alpha^\top v \geq \beta$ for all $v \in \mathcal{V}$. Furthermore, suppose for a contradiction $\alpha^\top r < 0$ for some $r \in \mathcal{R}$. For a fixed $v \in \mathcal{V}$, the point $v + \lambda r$ is feasible for any $\lambda \geq 0$ but $\alpha^\top (v + \lambda r)$ strictly decreases as λ increases, leading to a contradiction. □

Figure 2.2 provides a geometric interpretation of Theorem 2.12. The first condition states that all vertices must lie on or above the hyperplane, while the second condition states that the rays must point away from the hyperplane.

2.4 Facets of the Disjunctive Hull

In this section, we present a detailed proof of Theorem 1 from [4]. While the original paper outlines the main ideas, the argument is presented at a high level. Here, we provide a complete formalization of the result, along with related lemmas and supporting results. Given a pointed polyhedron $\mathcal{P} = \operatorname{conv}(\mathcal{V}) + \operatorname{cone}(\mathcal{R}) \subseteq \mathbb{R}^n$, we define

$$\mathcal{C} := \{(\alpha, \beta) \in \mathbb{R}^{n+1} \mid \alpha^\top v - \beta \geq 0 \text{ for all } v \in \mathcal{V} \text{ and } \alpha^\top r \geq 0 \text{ for all } r \in \mathcal{R}\}.$$

An immediate consequence of Theorem 2.12 is the following corollary which shows that $\mathcal{C}$ is the set of all valid inequalities for $\mathcal{P}$.

Corollary 2.13. Let $\mathcal{P} = \operatorname{conv}(\mathcal{V}) + \operatorname{cone}(\mathcal{R}) \subseteq \mathbb{R}^n$ be a pointed polyhedron admitting a $\mathcal{V}$-polyhedral representation. Then

$$\mathcal{P} = \{x \in \mathbb{R}^n \mid \alpha^\top x \geq \beta \text{ for all } (\alpha, \beta) \in \mathcal{C}\}.$$

Proof. We first show the inclusion from right to left. Consider the $\mathcal{H}$-polyhedral representation of $\mathcal{P}$. All inequalities in this representation are valid for $\mathcal{P}$ and are therefore in $\mathcal{C}$ by Theorem 2.12. Conversely, let $(\alpha, \beta) \in \mathcal{C}$. Then, by Theorem 2.12, the inequality $\alpha^\top x \geq \beta$ is valid for $\mathcal{P}$. Therefore, any point $x \in \mathcal{P}$ satisfies all such inequalities, and thus any x satisfying all inequalities in $\mathcal{C}$ must lie in $\mathcal{P}$. □

The set $\mathcal{C}$ is a polyhedral cone. In particular, if $(\alpha, \beta) \in \mathcal{C}$, then $(\lambda\alpha, \lambda\beta) \in \mathcal{C}$ for all $\lambda \geq 0$. However, we can truncate this cone by adding the restriction $\beta \in \{-1, 0, 1\}$ as suggested in [35]. We obtain the following corollary.

Corollary 2.14. Let $\mathcal{P} = \operatorname{conv}(\mathcal{V}) + \operatorname{cone}(\mathcal{R}) \subseteq \mathbb{R}^n$ be a pointed polyhedron admitting a $\mathcal{V}$-polyhedral representation. Then

$$\mathcal{P} = \{x \in \mathbb{R}^n \mid \alpha^\top x \geq \beta \text{ for all } (\alpha, \beta) \in \mathcal{C} \text{ with } \beta \in \{-1, 0, 1\}\}$$

Proof. The inclusion from left to right is analogous to Corollary 2.13. To show the inclusion from right to left, consider the $\mathcal{H}$-polyhedral representation of P. Each equality can be expressed as two inequalities. All less than or equal inequalities can be replaced with greater or equal inequalities by multiplying both sides by -1. Finally, greater or equal inequalities with non-zero right-hand side can be normalized by dividing both sides by $|\beta|$. □

Our next goal is to establish a correspondence between extreme rays of $\mathcal{C}$ and facet-defining inequalities of $\mathcal{P}$. We first show a lemma that states that $\mathcal{C}$ is a pointed polyhedral cone if $\mathcal{P}$ is full-dimensional.

Lemma 2.15. Let $\mathcal{P} = \operatorname{conv}(\mathcal{V}) + \operatorname{cone}(\mathcal{R}) \subseteq \mathbb{R}^n$ be a pointed polyhedron and $\mathcal{C}$ defined as above. If $\mathcal{P}$ is full-dimensional, then $\mathcal{C}$ is pointed.

Proof. Suppose for a contradiction $\mathcal{C}$ is not pointed, then $\mathcal{C}$ contains a line. Hence, there exists an $(\alpha, \beta) \neq 0$ such that $(\alpha, \beta) \in \mathcal{C}$ and $(-\alpha, -\beta) \in \mathcal{C}$. We have that $\alpha \neq 0$, since otherwise, for an arbitrary but fixed vertex v of the polyhedron, we have

$$\alpha^\top v - \beta = -\beta \geq 0$$

and

$$-\alpha^\top v - (-\beta) = \beta \geq 0$$

which imply $\beta = 0$, a contradiction. We now have that the inequality $\alpha^\top x \geq \beta$ and $-\alpha^\top x \geq -\beta$ are both valid for $\mathcal{P}$ by the construction of $\mathcal{C}$. Hence $\mathcal{P} \subseteq \{x \in \mathbb{R}^n \mid \alpha^\top x = \beta\}$ which contradicts the full-dimensionality of $\mathcal{P}$. □

Next, we consider the $\mathcal{V}$-polyhedral description of a face of $\mathcal{P}$

Lemma 2.16. Let $\mathcal{P} = \text{conv}(\mathcal{V})+\text{cone}(\mathcal{R}) \subseteq \mathbb{R}^n$ be a pointed polyhedron admitting a $\mathcal{V}$-polyhedral representation and $\{x \in \mathbb{R}^n \mid \alpha^\top x \geq \beta\}$ a half-space defining a face $\mathcal{F}$. Then

$$\mathcal{F} = \text{conv}(\mathcal{V}') + \text{cone}(\mathcal{R}')$$

where $\mathcal{V}' = \{v \in \mathcal{V} \mid \alpha^\top v = \beta\}$ and $\mathcal{R}' = \{r \in \mathcal{R} \mid \alpha^\top r = 0\}$.

Proof. Let $T = \text{conv}(\mathcal{V}') + \text{cone}(\mathcal{R}')$. We first show the inclusion from right to left. Since $\mathcal{V}' \subseteq \mathcal{V}$ and $\mathcal{R}' \subseteq \mathcal{R}$, we have $T \subseteq \mathcal{P}$.

Let $t \in T$, then

$$t = \sum_{i=1}^{m} \mu_i v_i + \sum_{i=1}^{n} \lambda_i r_i$$

where $v_1, \ldots, v_m \in \mathcal{V}'$, $r_1, \ldots, r_n \in \mathcal{R}'$, $\mu \geq 0$, $\lambda \geq 0$, $\sum_{i=1}^m \mu_i = 1$.

We have

$$\begin{aligned} \alpha^\top t &= \sum_{i=1}^{m} \mu_i \alpha^\top v_i + \sum_{i=1}^{n} \lambda_i \alpha^\top r_i \\ &= \sum_{i=1}^{m} \mu_i \beta = \beta. \end{aligned} \tag{2.3}$$

So $T \subseteq \{x \in \mathbb{R}^n \mid \alpha^\top x = \beta\}$, which implies $T \subseteq \mathcal{F}$.

We now show the inclusion from left to right. We first note that for the inequality $\alpha^\top x \geq \beta$ to be valid for $\mathcal{P}$ we must have $\alpha^\top r \geq 0$ for all $r \in \mathcal{R}$. Let $x \in \mathcal{F}$. Since $x \in \mathcal{P}$, we can write

$$x = \sum_{i=1}^{m} \mu_i v_i + \sum_{i=1}^{n} \lambda_i r_i$$

where $v_1, \ldots, v_m \in \mathcal{V}$, $r_1, \ldots, r_n \in \mathcal{R}$, $\mu \geq 0$, $\lambda \geq 0$, $\sum_{i=1}^m \mu_i = 1$.

We have

$$\alpha^\top x = \sum_{i=1}^{m} \mu_i \alpha^\top v_i + \sum_{i=1}^{n} \lambda_i \alpha^\top r_i \geq \sum_{i=1}^{m} \mu_i \beta = \beta.$$

Since $x \in \mathcal{F}$, we must also have $\alpha^\top x = \beta$. Equality holds only if $\mu_i = 0$ for all $i \in [m]$ with $\alpha^\top v_i > \beta$ and $\lambda_i = 0$ for all $i \in [n]$ with $\alpha^\top r_i > 0$. □

The intuition behind Lemma 2.16 is that when we are given a $\mathcal{V}$-polyhedron, the face defined by a face-defining inequality is the Minkowski sum of convex hull of the set of vertices that lie in the hyperplane corresponding to the inequality and the conic hull of all rays parallel to the inequality. As a visualization of this, observe that in Figure 2.2, the polyhedron $\mathcal{P}$ has three facets given by $\text{conv}(\{v_1, v_2\})$, $\text{conv}(\{v_1\}) + \text{cone}(\{r_1\})$, and $\text{conv}(\{v_2\}) + \text{cone}(\{r_2\})$

Our last lemma concerns the dimension of a $\mathcal{V}$-polyhedron. We state the lemma explicitly, but the proof follows directly from the definition of dimension.

Lemma 2.17. Let $\mathcal{P} = \text{conv}(\mathcal{V})+\text{cone}(\mathcal{R}) \subseteq \mathbb{R}^n$ be a pointed polyhedron admitting a $\mathcal{V}$-polyhedral representation and $v_0 \in \mathcal{V}$ arbitrary but fixed. Then

$$\dim(\mathcal{P}) = \dim(\text{span}(\{v - v_0 \mid v \in \mathcal{V}\} \cup \mathcal{R}))$$

We now prove Theorem 1 of [4] that the facet-defining inequalities of $\mathcal{P}$ correspond precisely to the extreme rays of $\mathcal{C}$.

Theorem 2.18. Let $\mathcal{P} = \text{conv}(\mathcal{V}) + \text{cone}(\mathcal{R}) \subseteq \mathbb{R}^n$ be a pointed, full-dimensional polyhedron, and let $\mathcal{C}$ be defined as above. An inequality $\alpha^\top x \geq \beta$ is facet-defining for $\mathcal{P}$ if and only if (α, β) is an extreme ray of $\mathcal{C}$.

Proof. Let $\alpha^\top x \geq \beta$ be a facet-defining inequality for $\mathcal{P}$. Consider the sets $\mathcal{V}'$ and $\mathcal{R}'$ as defined in Lemma 2.16, and fix an arbitrary $v_0 \in \mathcal{V}'$.

Since a facet has dimension $n-1$, Lemma 2.17 implies that we can choose at least $n-1$ linearly independent vectors from the set $\{v - v_0 \mid v \in \mathcal{V}\} \cup \mathcal{R}$. Let

$$v_1 - v_0, \dots, v_k - v_0,\ r_1, \dots, r_\ell$$

be the selected elements, where $k + \ell = n - 1$.

The vectors $v_0, v_1, \dots, v_k$ are affinely independent; hence, the lifted vectors $(v_0, -1), \dots, (v_k, -1)$ are linearly independent. Moreover, the lifted vectors $(r_1, 0), \dots, (r_\ell, 0)$ are also linearly independent. Therefore, combining these two sets yields n linearly independent vectors corresponding to constraints defining $\mathcal{C}$. Since all of these constraints are tight at (α, β) and $\mathcal{C}$ is pointed, it follows that (α, β) is an extreme ray of $\mathcal{C}$.

Conversely, let $(\alpha, \beta) \neq 0$ be an extreme ray of $\mathcal{C}$. Then $\alpha \neq 0$, since otherwise, for any fixed vertex $v \in \mathcal{V}$, we would have

$$\alpha^\top v - \beta = -\beta \geq 0 \quad \text{and} \quad -\alpha^\top v + \beta = \beta \geq 0,$$

which together imply $\beta = 0$, contradicting $(\alpha, \beta) \neq 0$.

Since $\mathcal{C}$ is pointed, there exist n linearly independent constraints active at (α, β). At least one of them corresponds to a vertex, because if $\alpha^\top r_i = 0$ for a set of n linearly independent rays $r_1, \dots, r_n$, then it would follow that $\alpha = 0$, again a contradiction.

Let

$$\begin{pmatrix} v_1 \\ -1 \end{pmatrix}, \dots, \begin{pmatrix} v_k \\ -1 \end{pmatrix}, \quad \begin{pmatrix} r_1 \\ 0 \end{pmatrix}, \dots, \begin{pmatrix} r_\ell \\ 0 \end{pmatrix}$$

be the selected rows with $k+\ell = n$. We claim that the set of vectors $\{v_2 - v_1, \dots, v_k - v_1, r_1, \dots, r_\ell\}$ is linearly independent. Suppose, for contradiction, that they are not. Then there exist scalars $\lambda_2, \dots, \lambda_n$, not all zero, such that

$$\sum_{i=2}^{k} \lambda_i (v_i - v_1) + \sum_{i=1}^{\ell} \lambda_{k+i} r_i = 0.$$

Let $\lambda_1 = -\sum_{i=2}^{k} \lambda_i$. Consider the lifted linear combination:

$$\lambda_1 \begin{pmatrix} v_1 \\ -1 \end{pmatrix} + \sum_{i=2}^{k} \lambda_i \begin{pmatrix} v_i \\ -1 \end{pmatrix} + \sum_{i=1}^{\ell} \lambda_{k+i} \begin{pmatrix} r_i \\ 0 \end{pmatrix}.$$

The first n components of this sum are:

$$\lambda_1 v_1 + \sum_{i=2}^{k} \lambda_i v_i + \sum_{i=1}^{\ell} \lambda_{k+i} r_i = \sum_{i=2}^{k} \lambda_i (v_i - v_1) + \sum_{i=1}^{\ell} \lambda_{k+i} r_i = 0,$$

The last component is:

$$-\lambda_1 - \sum_{i=2}^{k} \lambda_i = 0,$$

by the definition of λ_1. We obtain a contradiction to the assumption that the active constraints are linearly independent.

Finally, by Lemma 2.17, the face defined by the inequality $\alpha^\top x \geq \beta$ has dimension $n-1$ and thus defines a facet. □

2.5 Simple VPCs

By combining Theorem 2.10 and Theorem 2.12, we can now formulate for a given disjunctive set an LP whose feasible solutions correspond to valid inequalities for the disjunctive hull. We will call this LP the *point-ray linear program* (PRLP). Given a disjunctive set $\bigcup_{i=1}^{N} \{x \in \mathbb{R}^n \mid D_i x \leq d_i\}$ where each disjunctive term is nonempty, pointed and admits a $\mathcal{V}$-polyhedral representation

$$\{x \in \mathbb{R}^n \mid D_i x \leq d_i\} = \operatorname{conv}(\mathcal{V}_i) + \operatorname{cone}(\mathcal{R}_i),$$

a cost vector $c \in \mathbb{R}^n$, and a fixed $\beta \in \{-1, 0, 1\}$, the corresponding PRLP is given by

$$\begin{aligned} \min \quad & c^\top \alpha \\ \text{s.t} \quad & \alpha^\top v \geq \beta \text{ for all } v \in \bigcup_{i=1}^{N} \mathcal{V}_i \\ & \alpha^\top r \geq 0 \text{ for all } r \in \bigcup_{i=1}^{N} \mathcal{R}_i \\ & \alpha \in \mathbb{R}^n. \end{aligned}$$

The selection of the cost vector and β will be discussed in the next chapter. We normalize $\beta \in \{-1, 0, 1\}$, as the LP would otherwise be unbounded. As shown in Corollary 2.14, this normalization does not restrict the set of obtainable inequalities, since any inequality with general β can be obtained up to positive scaling.

Example 2.19. We use the MIP from Example 2.4 and consider the disjunctive set $\mathcal{D} = \mathcal{D}_1 \cup \mathcal{D}_2$ where

$$\mathcal{D}_1 := \mathcal{P} \cap \{x \in \mathbb{R}^2 \mid x_1 \leq 5, x_2 \leq 3\} \text{ and } \mathcal{D}_2 := \mathcal{P} \cap \{x \in \mathbb{R}^2 \mid x_1 \leq 4, x_2 \geq 4\}.$$

The disjunctive set is illustrated in panel (a) of Figure 2.3. The $\mathcal{V}$-polyhedral representation of the disjunctive terms are

$$\mathcal{D}_1 = \text{conv}\left(\left\{(1,1), \left(\frac{31}{7}, 1\right), \left(5, \frac{7}{3}\right), (5,3), (1,3)\right\}\right)$$

and

$$\mathcal{D}_2 = \text{conv}\left(\{(1,4), (4,4), (4, 4.25), (2.5, 5), (1, 4.4)\}\right)$$

For a fixed $\beta \in \{-1, 0, 1\}$ and a cost vector $c \in \mathbb{R}^n$, the PRLP is

$$\begin{aligned} \min \quad & c_1\,\alpha_1 + c_2\,\alpha_2 \\ \text{s.t.} \quad & \alpha_1 + \alpha_2 \geq \beta \\ & \tfrac{31}{7}\,\alpha_1 + \alpha_2 \geq \beta \\ & 5\,\alpha_1 + \tfrac{7}{3}\,\alpha_2 \geq \beta \\ & 5\,\alpha_1 + 3\,\alpha_2 \geq \beta \\ & \alpha_1 + 3\,\alpha_2 \geq \beta \\ & \alpha_1 + 4\,\alpha_2 \geq \beta \\ & 4\,\alpha_1 + 4\,\alpha_2 \geq \beta \\ & 4\,\alpha_1 + 4.25\,\alpha_2 \geq \beta \\ & 2.5\,\alpha_1 + 5\,\alpha_2 \geq \beta \\ & \alpha_1 + 4.4\,\alpha_2 \geq \beta \\ & \alpha_1, \alpha_2 \in \mathbb{R}. \end{aligned}$$

There are two drawbacks of methods based on the $\mathcal{V}$-polyhedral representations of the disjunctive hull: first, the $\mathcal{V}$-polyhedral representation of each term is in general unknown, and second, the $\mathcal{V}$-polyhedral representation for a polyhedron may in some cases be exponential in comparison to the number of inequalities in the $\mathcal{H}$-polyhedral representation. The latter implies that we might possibly have exponentially many constraints in our PRLP compared to the $\mathcal{H}$-polyhedral description of the disjunctive set. A classic example of this is the unit cube in $\mathbb{R}^n$, which is defined by $2n$ inequalities but has 2^n vertices. In this section, we introduce one approach to circumvent these problems presented first in [4]. An alternative approach using row-generation is proposed in [36] and [37].

Our approach is based on the following lemma. We recall that we denote the disjunctive hull of a disjunctive set $\mathcal{D}$ by $\bar{\mathcal{D}}$.

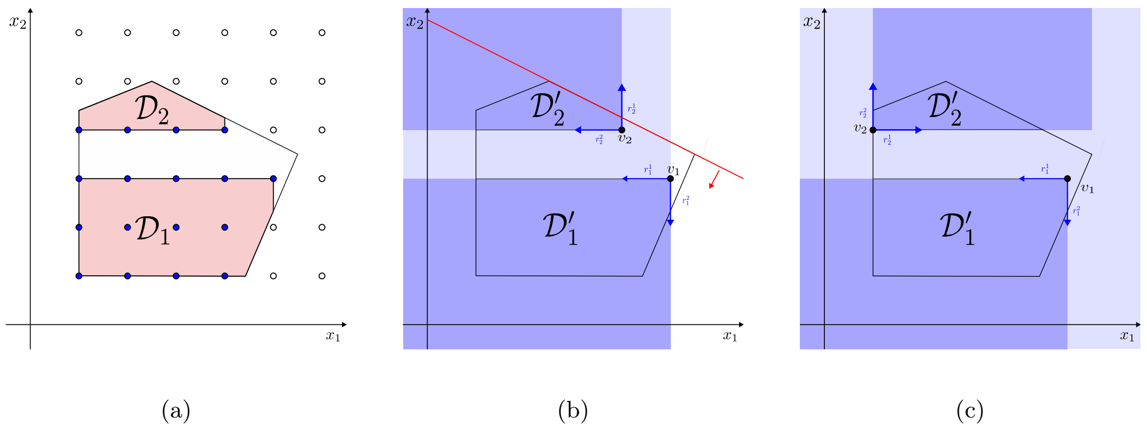

Figure 2.3: Figure for Example 2.19. Panel (a) shows the disjunctive set $\mathcal{D}_1 \cup \mathcal{D}_2$. Panel (b) shows the relaxed disjunctive set $\mathcal{D}_1' \cup \mathcal{D}_2'$, obtained by taking the corner polyhedron corresponding to the vertex $v_1 = (5,3)$ from $\mathcal{D}_1$ and $v_2 = (4,4)$ from $\mathcal{D}_2$, its disjunctive hull, along with an inequality valid for the disjunctive hull of the original set but not the relaxed set. Panel (c) shows the relaxed disjunctive set $\mathcal{D}_1' \cup \mathcal{D}_2'$, obtained by taking the corner polyhedron corresponding to the vertex $v_1 = (5,3)$ from $\mathcal{D}_1$ and $v_2 = (1,4)$ from $\mathcal{D}_2$, along with its disjunctive hull.

Lemma 2.20. Given a disjunctive set $\bigcup_{i=1}^N \mathcal{D}_i$ and relaxations $\mathcal{D}_i' \supseteq \mathcal{D}_i$ where each relaxation is pointed, non-empty, and admits a $\mathcal{V}$-polyhedral representation

$$\mathcal{D}_i' = \operatorname{conv}(\mathcal{V}_i) + \operatorname{cone}(\mathcal{R}_i).$$

We define the relaxed disjunctive set $\mathcal{D}' := \cup_{i=1}^N \mathcal{D}_i'$. An inequality $\alpha^\top x \geq \beta$ is valid for $\bar{\mathcal{D}}'$ if and only if the following conditions are satisfied:

(i) $\alpha^\top v \geq \beta$ for all $v \in \bigcup_{i=1}^N \mathcal{V}_i$,

(ii) $\alpha^\top r \geq 0$ for all $r \in \bigcup_{i=1}^N \mathcal{R}_i$.

Furthermore, any valid inequality for $\bar{\mathcal{D}}'$ is also valid for $\bar{\mathcal{D}}$.

Proof. To show the first statement, apply Theorem 2.12 to the set $\bar{\mathcal{D}}'$. The second statement follows from the construction of D_i' for $i \in [N]$ and the definition of valid inequality. □

Given a disjunctive set $\bigcup_{i=1}^N \mathcal{D}_i$, our goal is to find a relaxation $\mathcal{D}_i' \supseteq \mathcal{D}_i$ for each $i \in [N]$ with a simple $\mathcal{V}$-polyhedral representation. One relaxation that we can use is to pick for each $i \in [N]$, an arbitrary vertex and take as the relaxed set $\mathcal{D}_i'$ the corner polyhedron defined by that vertex. We call this relaxed disjunctive set $\mathcal{D}' = \cup_{i=1}^N \mathcal{D}'$ the *corner polyhedron relaxation*. We denote the disjunctive hull of this set with $\bar{\mathcal{D}}'$. $\mathcal{V}$-polyhedral cut obtained from the corner polyhedron relaxation is often referred to in the literature as *simple VPC*. Corner polyhedron relaxations are convenient because each disjunctive term in the relaxation can be described by a single vertex and n rays where n is the dimension of the problem.

Example 2.21. We continue our discussion from Example 2.19. We select the vertex $v_1 = (5,3)$ from $\mathcal{D}_1'$ and $v_2 = (4,4)$ from $\mathcal{D}_2'$. The corner polyhedra corresponding to these vertices are denoted by $\mathcal{D}_1'$ and $\mathcal{D}_2'$ respectively.

We obtain the corner polyhedron relaxation $\mathcal{D}' = \mathcal{D}_1' \cup \mathcal{D}_2'$. We illustrate this set along with $\bar{\mathcal{D}}'$ in panel (b) of Figure 2.3. It is easy to see that any valid inequality for the $\bar{\mathcal{D}}'$ is also valid for $\bar{\mathcal{D}}$. However, the converse does not hold. For example, the inequality $2x_1 + 4x_2 \leq 25$, colored red, is valid for $\bar{\mathcal{D}}$ but not $\bar{\mathcal{D}}'$.

A worse corner polyhedron relaxation is obtained when we select $v_2 = (1,4)$ while keeping v_1 as illustrated in panel (c) of Figure 2.3. The disjunctive hull of the relaxed set is $\mathbb{R}^2$. Hence, we see that when constructing the corner polyhedron relaxation the choice of vertex that we use is important.

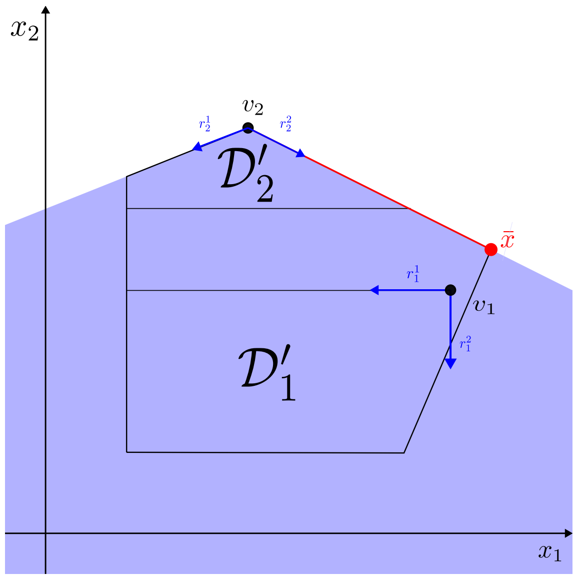

Figure 2.4: Figure for Example 2.22. The disjunctive hull of the corner polyhedron relaxation, obtained by taking $v_1 = (5,3)$ and $v_2 = (2.5,5)$, is shown in blue. The optimal facet is highlighted in red, and the LP optimal solution is marked as a red point.

Given a MIP and a valid disjunction $\mathcal{D}$ such that $\bar{x} \notin \bar{\mathcal{D}}$ where $\bar{x}$ is an optimal solution to the LP relaxation of the MIP. We want to select a vertex from each disjunctive term such that the resulting corner polyhedron relaxation $\mathcal{D}'$ satisfies $\bar{x} \notin \bar{\mathcal{D}}'$. In [4], the choice of vertex for each disjunctive term is the one that maximize the objective function of the MIP, that is, v_i is an optimal vertex to $\min\{c^\top x \mid x \in \mathcal{D}_i\}$ for each $i \in [N]$ where c is the cost vector of the MIP. While this approach is logical, as it accounts for the behavior of the MIP objective function for each disjunctive term, Example 2.22 shows that it can lead to a relaxation that contains $\bar{x}$.

Example 2.22. We consider the MIP from Example 2.19 with the modified objective function $2x_1 + 4x_2$. The set of optimal solution to the LP relaxation is a facet of the set of LP feasible solution $\mathcal{P}$. This is illustrated in Figure 2.4. The optimal facet is colored red, an LP optimal solution is marked as a red dot. Note that $v_1 = (5,3)$ and $v_2 = (2.5,5)$ are optimal vertices when optimizing the cost function over $\mathcal{D}_1$ and $\mathcal{D}_2$, respectively. However as illustrated in the figure the disjunctive hull of corner polyhedron relaxation contains $\bar{x}$.

We now state a condition under which we can guarantee that $\bar{x}$ is not in the disjunctive hull of the corner polyhedron relaxation.

Theorem 2.23. Let $\cup_{i=1}^N \mathcal{D}_i \subseteq \mathbb{R}^n$ be a disjunctive set, $\bar{x} \in \mathbb{R}^n$ a point, and $c \in \mathbb{R}^n$ an arbitrary vector. Furthermore, for each $i \in [N]$, let

$$\mathcal{D}_i' := \operatorname{conv}(\{v_i\}) + \operatorname{cone}(\{r_i^1, \ldots, r_i^n\})$$

be the corner polyhedron relaxation obtained by selecting an optimal vertex of the LP $\min\{c^\top x \mid x \in \mathcal{D}_i\}$. If $c^\top \bar{x} < c^\top v_i$ for all $i \in [N]$, then $\bar{x} \notin \bar{\mathcal{D}}'$ and PRLP is feasible.

Proof. Let $\epsilon > 0$ be small enough such that $c^\top \bar{x} + \epsilon \leq c^\top v^i$ for $i \in [N]$. By construction of the corner polyhedron, $c^\top r_i^k \geq 0$ for $i \in [N]$ and $k \in [N]$. So letting $\alpha = c$ and $\beta = c^\top \bar{x} + \epsilon$ gives a feasible solution to PRLP. Furthermore, $\{x \in \mathbb{R}^n \mid c^\top x = c^\top \bar{x} + \epsilon\}$ is a hyperplane strictly separating $\bar{x}$ from $\bar{\mathcal{D}}'$. □

The theorem is a reformulation of [4, Proposition 4] for general point and cost vector. Based on this theorem, we propose an alternative approach to select the corner polyhedron generating vertices that guarantee $\bar{x} \notin \bar{\mathcal{D}}'$. Our approach relies on the fact that a point in a polyhedron is a vertex if and only if it is the unique minimizer for some cost vector. In the theorem below, we will state this cost vector explicitly.

Theorem 2.24. Given a MIP of the form (2.1), we consider its LP relaxation and transform it by adding slack variables into the following equivalent LP

$$\begin{aligned} \min \quad & c^\top x \\ \text{s.t} \quad & Ax + Is = 0 \\ & -U \leq s \leq -L \\ & \ell \leq x \leq u \\ & x \in \mathbb{R}^n \\ & s \in \mathbb{R}^m. \end{aligned} \tag{2.4}$$

If (x, s) is a basic feasible solution to the LP (2.4), then (x, s) is the unique minimizer to the cost vector (c_x, c_s) given by

$$c_{x_i} = \begin{cases} -1, & \text{if } x_i = u_i, \\ 1, & \text{if } x_i = \ell_i, \\ 0, & \text{if } x_i \text{ is basic,} \end{cases}$$

for $i \in [n]$ and

$$c_{s_i} = \begin{cases} -1, & \text{if } s_i = -L_i, \\ 1, & \text{if } s_i = -U_i, \\ 0, & \text{if } s_i \text{ is basic,} \end{cases}$$

for $i \in [m]$.

Proof. Observe that basic variables do not affect the cost. For the nonbasic variables, moving away from the bounds fixed in (x, s) strictly increases the cost. The minimizer is unique because the basic variables are uniquely determined by the nonbasic variables and the equality $Ax+Is = 0$. $\square$

We can rewrite the cost vector such that only the non-slack variables have nonzero cost. We will denote with $(\bar{x}, \bar{s})$ the given basic feasible solution. Further, we denote with $\mathcal{N}_u$ the set of non-basic variables fixed to their upper-bound and $\mathcal{N}_\ell$ the set of non-basic variables fixed to their lower bound.

For a set $\mathcal{S}$, we denote by $\mathbb{1}_{\mathcal{S}}(x)$ the indicator function of $\mathcal{S}$. We compute

$$\begin{aligned} c_x^\top x + c_s^\top s &= \sum_{x_i \in \mathcal{N}_\ell} x_i - \sum_{x_i \in \mathcal{N}_u} x_i + \sum_{s_i \in \mathcal{N}_\ell} s_i - \sum_{s_i \in \mathcal{N}_u} s_i \\ &= \sum_{x_i \in \mathcal{N}_\ell} x_i - \sum_{x_i \in \mathcal{N}_u} x_i + \sum_{j=1}^{m} \mathbb{1}_{\mathcal{N}_\ell}(s_j) \sum_{i=1}^{n} a_{ij} x_i - \sum_{j=1}^{m} \mathbb{1}_{\mathcal{N}_u}(s_j) \sum_{i=1}^{n} a_{ij} x_i \\ &= \sum_{i=1}^{n} \left(\mathbb{1}_{\mathcal{N}_\ell}(x_i) - \mathbb{1}_{\mathcal{N}_u}(x_i) + \sum_{j=1}^{m} \mathbb{1}_{\mathcal{N}_\ell}(s_j) a_{ij} - \sum_{j=1}^{m} \mathbb{1}_{\mathcal{N}_u}(s_j) a_{ij} \right) x_i \end{aligned}$$

where we use $Ax + Is = 0$ for the first equality.

The last equation expresses the cost as a function of only the structural variables of the problem.

Chapter 3

Implementation of $\mathcal{V}$-Polyhedral Disjunctive Cuts

In this section, we highlight aspects of our implementation that are non-trivial, insightful, or pedagogically useful. Each subsection focuses on a different aspect of our algorithms. We begin in Section 3.1 with a brief introduction to SCIP, the branch-and-bound framework used in our implementation. Next, we introduce in Section 3.2 the nonbasic space to allow us to find inequalities that are violated by an optimal solution to the LP relaxation of the MIP. We also discuss how the rays from each corner polyhedron can be collected efficiently. Next, in Section 3.3, we discuss how LP information can be obtained from the SCIP API to give the reader an intuition of how theory translates to implementation. Next, we explain in Section 3.4 how we obtain disjunctions from a partial branch-and-bound tree. Relevant details on the implementation of the algorithm are provided in the same section. Finally, in Section 3.5, we discuss the choice of objective vector for the cut generation LP.

3.1 SCIP

SCIP [1] is an open-source solver for mixed-integer programs (MIPs) that originated as the PhD project of Tobias Achterberg [38]. It is distributed under the Apache 2.0 license. It supports general constraint integer programs; however, we primarily use it as a branch-and-bound MIP solver.

We choose SCIP for three main reasons. First, although its overall MIP performance still lags behind leading commercial solvers such as Xpress [2], it consistently outperforms other open-source alternatives like Cbc [6] (see the computational results in [7]). Second, because it is open source, SCIP exposes every stage of the branch-and-bound process, including presolving, heuristics, and cutting planes. This level of transparency is not available when using commercial solvers. Finally, SCIP is highly extensible: users can create plugins to handle new constraint types, implement custom primal heuristics, and—crucially for our work—add new cutting planes. Though originally written in C, SCIP provides interfaces to several other languages, including Python, Rust, and Julia. We remark that the SCIP documentation refers to cutting-plane plugins as *separators*.

To add a separator in SCIP, a user must define a callable object which, when invoked, adds cutting planes to SCIP. At each node of the branch-and-bound tree, SCIP first solves the LP relaxation of the node and then calls all separators available at that node. One cycle of this loop is often called a *separation round*. After each separator adds its cuts, SCIP reoptimizes the LP relaxation. This process repeats until a termination criterion is met.

This separation process can be customized in several ways. First, we can assign priorities to separator objects to determine the order in which they are called. These priorities are integers, with separators of higher priority being called earlier. Second, we can assign a frequency to a separator: a frequency of -1 means the separator is never called; a frequency of 0 means it is called only at the root node; and a positive frequency k means the separator is called at every node at depth nk, where $n \in \mathbb{N}$.

3.2 Working In The Non-Basic Space

In the previous chapter, we discussed how to find valid inequalities for a given disjunctive hull. We now extend the theory to find a cutting plane that is violated by the optimal solution of the LP relaxation of a MIP. We first summarize our setup thus far. We are given a MIP and an optimal solution to the LP relaxation $\bar{x}$. Furthermore, we are given a disjunction $\mathcal{D} := \cup_{i=1}^{N} \mathcal{D}_i$ with $\mathcal{P}_I \in \bar{\mathcal{D}}$ and $\bar{x} \notin \mathcal{D}$. Our goal is to find an inequality $\alpha^\top x \geq \beta$ such that $\alpha^\top \bar{x} < \beta$ and $\alpha^\top x \geq \beta$ for all $x \in \bar{\mathcal{D}}$. The latter implies $\alpha^\top x \geq \beta$ for all $x \in \mathcal{P}_I$. Hence, $\alpha^\top x \geq \beta$ is a cutting plane for our MIP.

To avoid dealing with an exponential number of rays and vertices, we use as a proxy the corner polyhedron relaxation of our disjunction $\mathcal{D}' := \cup_{i=1}^{N} \mathcal{D}'_i$. Each term of this relaxation has a simple $\mathcal{V}$-polyhedral representation given by

$$\mathcal{D}'_i = \operatorname{conv}(\{v_i\}) + \operatorname{cone}(\{r_i^1, \ldots, r_i^n\}).$$

To find an inequality $\alpha^\top x \geq \beta$ that is valid for $\mathcal{D}'$, we can solve the PRLP

$$\begin{aligned} \min \quad & c^\top \alpha \\ \text{s.t.} \quad & \alpha^\top v \geq \beta \text{ for all } v \in \{v_1, \ldots, v_N\}, \\ & \alpha^\top r \geq 0 \text{ for all } r \in \bigcup_{i=1}^{N} \{r_i^1, \ldots, r_i^n\}, \\ & \alpha \in \mathbb{R}^n. \end{aligned}$$

where $c \in \mathbb{R}^n$ and $\beta \in \{-1, 0, 1\}$ are given hyperparameters.

In order to restrict ourselves to inequalities that cut off $\bar{x}$, we consider the translated space where $\bar{x}$ lies at the origin. This space is often referred to in the literature as the *non-basic space.* We denote the $\mathcal{V}$-polyhedral representation of $\mathcal{D}'_i$ for $i \in [N]$ in this space with

$$\mathcal{D}'_i = \operatorname{conv}(\{v'_i\}) + \operatorname{cone}(\{r_i^1, \ldots, r_i^n\}),$$

where $v'_i = v_i - \bar{x}$. Furthermore, we fix $\beta = 1$. The PRLP becomes

$$\begin{aligned} \min \quad & c^\top \alpha \\ \text{s.t.} \quad & \alpha^\top (v - \bar{x}) \geq 1 \text{ for all } v \in \{v_1, \ldots, v_n\}, \\ & \alpha^\top r \geq 0 \text{ for all } r \in \bigcup_{i=1}^{N} \{r_i^1, \ldots, r_i^n\}, \\ & \alpha \in \mathbb{R}^n. \end{aligned} \tag{$*$}$$

where $c \in \mathbb{R}^n$ is a given hyperparameter. The following theorem states that any feasible solution to this modified PRLP corresponds (up to a scalar) to an inequality that is valid for $\mathcal{D}'$ but violated by $\bar{x}$.

Theorem 3.1. In the setup described above, let $\alpha^\top x \geq \beta$ be a valid inequality for $\mathcal{D}'$ with $\alpha^\top \bar{x} < \beta$. Then there exist $\lambda > 0$ and a feasible solution $\bar{\alpha} \in \mathbb{R}^n$ to $(*)$ such that $\lambda \bar{\alpha} = \alpha$ and $\lambda(\bar{\alpha}^\top \bar{x} + 1) = \beta$.

Proof. Let $\alpha^\top x \geq \beta$ be such an inequality. We define $\lambda := \beta - \alpha^\top \bar{x}$ and $\bar{\alpha} := \alpha / \lambda$. Clearly, $\lambda > 0$. Since $\alpha^\top x \geq \beta$ is valid for $\mathcal{D}'$, we have

$$\bar{\alpha}^\top (v_i - \bar{x}) = \frac{1}{\lambda}(\alpha^\top v_i - \alpha^\top \bar{x}) \geq \frac{1}{\lambda}(\beta - \alpha^\top \bar{x}) = 1$$

for all $i \in [N]$. Furthermore, by Theorem 2.12, we have $\alpha^\top r_i^j \geq 0$ for all $i \in [N]$ and $j \in [n]$, which implies $\bar{\alpha}^\top r_i^j \geq 0$. Hence, $\bar{\alpha}$ is a feasible solution to $(*)$. Finally, we have $\lambda\bar{\alpha} = \alpha$ and

$$\lambda(\bar{\alpha}^\top \bar{x} + 1) = \alpha^\top \bar{x} + \lambda = \beta,$$

which concludes the proof. □

Fixing β to 1 corresponds to adding the requirement that the generated cuts must be violated by $\bar{x}$. It is easy to see that had we fixed $\beta = 0$ in the construction of $(*)$, we would obtain the set of inequalities that pass through $\bar{x}$. The converse of Theorem 3.1 is also true.

Theorem 3.2. Let α be a feasible solution to $(*)$. Then $\alpha^\top x \geq \alpha^\top \bar{x} + 1$ is an inequality that is valid for $\mathcal{D}'$ and violated by $\bar{x}$.

Proof. Clearly, $\bar{x}$ violates the given inequality. To see that the inequality is valid for $\mathcal{D}'$, observe that the constraints of $(*)$ ensure that α and $\beta := \alpha^\top \bar{x} + 1$ satisfy the conditions in Theorem 2.12 applied to the disjunctive hull $\bar{\mathcal{D}}'$. □

We now introduce a secondary aspect we need to consider when working in the non-basic space. In our discussion of the simplex algorithm and corner polyhedron relaxations, we require that the LP relaxation of the MIP be in standard form. Given a MIP of the form

$$\begin{aligned} \min \quad & c^\top x \\ \text{s.t.} \quad & L \leq Ax \leq U, \\ & \ell \leq x \leq u, \\ & x_i \in \mathbb{Z} \quad (\forall i \in \mathcal{I}), \\ & x_i \in \mathbb{R} \quad (\forall i \in [n] \setminus \mathcal{I}), \end{aligned} \tag{3.1}$$

where $c \in \mathbb{R}^n$, $A \in \mathbb{R}^{m \times n}$ has full row rank, $L, \ell \in (\mathbb{R} \cup \{-\infty\})^m$, $U, u \in (\mathbb{R} \cup \{+\infty\})^n$, and $\mathcal{I} \subseteq [n]$, SCIP internally transforms the MIP into the equivalent formulation:

$$\begin{aligned} \min \quad & c^\top x \\ \text{s.t.} \quad & Ax + Is = 0, \\ & -U \leq s \leq -L, \\ & \ell \leq x \leq u, \\ & x_i \in \mathbb{Z} \quad (\forall i \in \mathcal{I}), \\ & x_i \in \mathbb{R} \quad (\forall i \in [n] \setminus \mathcal{I}), \\ & s \in \mathbb{R}^m. \end{aligned} \tag{3.2}$$

Although this system is not exactly in standard form as we have defined it, the presence of a full-rank system of linear equalities enables our construction of a simplex tableau, and hence also the construction of corner polyhedron relaxations. A crucial difference, however, is that since the variables of the transformed problem have two-sided bounds, they can be fixed to either bound. Hence, extra care is needed. In particular, if a non-basic variable is fixed to the upper bound, the ray corresponding to that non-basic variable should correspond to decreasing the non-basic variable. For reasons we will state later in Section 3.5, and to simplify notation for the rest of this section, we replace such variables with their negative complements in all of the resulting tableaux we acquire. This allows rays to always correspond to increasing a non-basic variable.

The introduction of the m slack variables increases the number of variables from n to $n + m$. Consequently, the dimension of $(*)$ is also $n + m$. We now outline a technique based on projection that reduces this dimension back to n.

From this point onward, for ease of notation, we distinguish the structural (original) variables from the slack variables by their indices. With a slight abuse of notation, we redefine

$$x := (x, s), \quad c := (c, 0), \quad \ell := (\ell, -U), \quad u := (u, -L), \quad A := \begin{bmatrix} A & I \end{bmatrix}.$$

Hence, we rewrite the transformed MIP as

$$\begin{aligned} \min \quad & c^\top x \\ \text{s.t.} \quad & Ax = 0, \\ & \ell \leq x \leq u, \\ & x_i \in \mathbb{Z} \quad \text{for all } i \in \mathcal{I}, \\ & x_i \in \mathbb{R} \quad \text{for all } i \in [n] \setminus \mathcal{I}. \end{aligned} \tag{3.3}$$

We denote the j-th column of the new matrix A by A_j for every $j \in [n+m]$.

We assume that, in addition to $\bar{x}$, we are given an optimal basis $\mathcal{B}$ for our LP relaxation. Recall that we denote the submatrices of A corresponding to the basic and non-basic variables by B and N, respectively, and the basic and non-basic variables by x_B and x_N, respectively. We denote by π_N the projection map onto the non-basic variables. We also define a lifting function σ_N, which lifts a vector from the projected space to the original space by appending it with zeros.

We define our computational form of PRLP

$$\begin{aligned} \min \quad & c^\top \alpha \\ \text{s.t.} \quad & \alpha^\top (\pi_N(v) - \pi_N(\bar{x})) \geq 1 \text{ for all } v \in \{v_1, \dots, v_N\}, \\ & \alpha^\top \pi_{\mathcal{N}}(r) \geq 0 \text{ for all } r \in \bigcup_{i=1}^{N} \{r_i^1, \dots, r_i^n\}, \\ & \alpha \in \mathbb{R}^n. \end{aligned} \tag{PRLP}$$

The relation between $(*)$ and (PRLP) is given by the following theorem.

Theorem 3.3. If α is a solution of (PRLP), then the lifted vector $\sigma_N(\alpha)$ is a solution of $(*)$.

Proof. We have for all $i \in [N]$

$$\sigma_{\mathcal{N}}(\alpha)^\top (v_i - \bar{x}) = \alpha^\top \pi_N(v_i - \bar{x}) \geq 1.$$

We also have for all $i \in [N]$ and $j \in [n]$

$$\sigma_{\mathcal{N}}(\alpha)^\top r_i^j = \alpha^\top \pi_{\mathcal{N}}(r_i^j) \geq 0.$$

□

We now present an argument that (PRLP) does not reduce the set of possible cuts that we can obtain compared to $(*)$. We denote the set of LP-feasible solutions to our transformed MIP by $\mathcal{P}$. We say two inequalities $\alpha_1^\top x \geq \beta_1$ and $\alpha_2^\top x \geq \beta_2$ are equivalent with respect to $\mathcal{P}$ if

$$\mathcal{P} \cap \{x \in \mathbb{R}^{n+m} \mid \alpha_1^\top x \geq \beta_1\} = \mathcal{P} \cap \{x \in \mathbb{R}^{n+m} \mid \alpha_2^\top x \geq \beta_2\}.$$

For a given $\alpha \in \mathbb{R}^n$, we define the *support* of α as

$$\operatorname{supp}(\alpha) = \{i \in [n] \mid \alpha_i \neq 0\}.$$

Theorem 3.4. Let $\alpha^\top x \geq \beta$ be an inequality with $\alpha \in \mathbb{R}^{n+m}$. There exists an equivalent inequality $\bar{\alpha}^\top x \geq \beta$ with respect to $\mathcal{P}$ such that $\operatorname{supp}(\bar{\alpha}) \subseteq \mathcal{N}$. In the special case that

$$\mathcal{D}' \subseteq \{x \in \mathbb{R}^{n+m} \mid Ax = 0\},$$

$\beta = 1$, and α is a feasible solution to $(*)$, then $\pi(\bar{\alpha})$ is a feasible solution to (PRLP).

Proof. Recall that we can rewrite basic variables in terms of non-basic variables, that is,

$$Bx_B + Nx_N = 0 \Leftrightarrow x_B = -B^{-1}Nx_N.$$

We define

$$\bar{\alpha}_i := \begin{cases} 0, & \text{if } i \in B, \\ \alpha_i - \alpha_B^\top B^{-1} A_i, & \text{if } i \in N. \end{cases}$$

We note that $\bar{\alpha}_N = \alpha_N - \alpha_B^\top B^{-1} A_N$. For any $p \in \mathcal{P}$, we have $p_B = -B^{-1} N p_N$, so

$$\begin{aligned} \alpha^\top p \geq \beta &\Leftrightarrow \alpha_B^\top p_B + \alpha_N^\top p_N \geq \beta \\ &\Leftrightarrow \bar{\alpha}_N^\top p_N \geq \beta \\ &\Leftrightarrow \bar{\alpha}^\top p \geq \beta, \end{aligned}$$

which shows that $\alpha^\top x \geq \beta$ and $\bar{\alpha}^\top x \geq \beta$ are equivalent with respect to $\mathcal{P}$.

Conversely, let α be a feasible solution to $(*)$. Since $\mathcal{D}' \subseteq \{x \in \mathbb{R}^{n+m} \mid Ax = 0\}$, for an arbitrary but fixed $v \in \{v_1, \dots, v_N\}$, we have $v_{\mathcal{B}} = B^{-1} N v_N$. Hence,

$$\begin{aligned} \pi_N(\bar{\alpha})^\top (\pi_N(v) - \pi_N(\bar{x})) &= \bar{\alpha}_N^\top (v_N - \bar{x}_N) \\ &= \alpha^\top (v - \bar{x}) \\ &\geq 1. \end{aligned}$$

We also have $Ar = 0$ for $r \in \bigcup_{i=1}^N \{r_i^1, \dots, r_i^n\}$, hence $r_{\mathcal{B}} = B^{-1} N r_N$. It follows that

$$\pi_N(\bar{\alpha})^\top \pi_N(r) = \bar{\alpha}_N^\top r_N = \alpha^\top r \geq 0.$$

Thus, $\pi_N(\bar{\alpha})$ is a feasible solution to (PRLP). $\square$

The two theorems show that we do not lose anything by working exclusively with (PRLP), which we will do.

A significant part of the practical performance of VPCs can be attributed to how the data for (PRLP) is collected. Obtaining the set of vertices, translating, and projecting them is relatively easy since there are only N vertices. Collecting the rays requires a bit more ingenuity. Done naively, we would need to gather nN rays, each of length $m + n$, before the projection. However, many of the ray entries will be zero. We now analyze which entries are nonzero.

At the beginning of the cut generation procedure, we have sets of indices that are basic $\mathcal{B}_{\bar{x}}$ and non-basic $\mathcal{N}_{\bar{x}}$ at $\bar{x}$. Next, we go over each disjunctive term $\mathcal{D}_i$ and optimize an LP over $\mathcal{D}_i$ to obtain an optimal vertex v, along with sets of indices that are basic $\mathcal{B}_v$ and non-basic $\mathcal{N}_v$ at v. Then, we need to collect the rays that define the corner polyhedron at vertex v. Recall that for an index $i \in \mathcal{N}_v$, the ray corresponding to the non-basic variable x_i is given componentwise by

$$r_k^i := \begin{cases} a_{k,i}, & \text{if } k \in \mathcal{B}_v, \\ 1, & \text{if } k = i, \\ 0, & \text{otherwise.} \end{cases}$$

where $a_{k,i}$ denotes the entry in the kth row and the ith column of the simplex tableau.

Since we project this ray to $\mathcal{N}_{\bar{x}}$, we can conclude that the only nonzero components of the projection of a ray r^i are the indices in $\mathcal{N}_{\bar{x}} \cap \mathcal{B}_v$, along with the ith component if $i \in \mathcal{N}_{\bar{x}}$ (with a value of 1). In other words, the number of nonzero entries is at most the number of non-basic variables in the original space that become basic when we optimize over the disjunction, plus one. Since at each iteration at most one non-basic variable can become basic, it is observed in [4] that the number of nonzero entries roughly corresponds to the number of simplex pivots during the reoptimization over $\mathcal{D}_i$. In our preliminary testing, this gives a significant speed-up in comparison to checking all indices of $\mathcal{B}_v$ in a naive implementation.

3.3 Obtaining LP Information From The SCIP LP Interface

We now describe how we can retrieve LP information in SCIP. This aims to help the reader understand how we proceed from theory to the implementation of cutting planes. As we mentioned before, SCIP allows us to define a callable object that will be invoked after solving the LP relaxation. SCIP provides API calls for accessing the LP data. As a result, the end user no longer needs to care about which LP solver is being used with SCIP. Our goal is to obtain the optimal tableau, that is, the rewritten version of our problem at the optimum. As mentioned earlier, SCIP internally transforms our problem into the form 3.2. For clarity, we restate the LP relaxation here:

$$\begin{aligned} \min \quad & c^\top x \\ \text{s.t.} \quad & Ax + Is = 0, \\ & -U \leq s \leq -L, \\ & \ell \leq x \leq u. \end{aligned} \tag{3.4}$$

We now have all the components to translate the vocabulary from the previous discussion to SCIP. An LP in SCIP is represented by a set of `SCIP_ROW` and `SCIP_COL` objects. Each `SCIP_COL` object corresponds to a structural variable x_i where $i \in [n]$, and each `SCIP_ROW` object corresponds to a slack variable s_i where $i \in [m]$. A basis matrix B, according to the SCIP LP Interface documentation, is an $m \times m$ invertible submatrix of the transformed constraint matrix $\begin{bmatrix} A & I \end{bmatrix}$. Hence, the basic variables consist of a mix of structural and slack variables. Each `SCIP_ROW` and `SCIP_COL` object has a basis status attribute that can be queried using `SCIProwGetBasisStatus` and `SCIPcolGetBasisStatus`, respectively. The basis status is `SCIP_BASESTAT_BASIC` if the corresponding structural or auxiliary variable is basic. Non-basic structural variables are fixed either to their upper bound, lower bound, or zero (if the variable is unbounded), corresponding to basis statuses `SCIP_BASESTAT_UPPER`, `SCIP_BASESTAT_LOWER`, and `SCIP_BASESTAT_ZERO`, respectively. Non-basic slack variables are marked based on whether the corresponding row is tight at the upper or lower bound. Consequently, if a row is tight at the upper bound, the basis status of the corresponding `SCIP_ROW` object is `SCIP_BASESTAT_UPPER`. This implies that the slack is tight at the lower bound of $-U$ in the transformed formulation, which is slightly counterintuitive. The reverse applies to rows that are tight at the lower bound.

The system of equalities used for the corner polyhedron relaxation of the optimal tableau can then be retrieved as follows: First, we apply the transformation

$$Ax + Is = 0 \Leftrightarrow B^{-1}Ax + B^{-1}s = 0.$$

SCIP provides the API calls `SCIPgetLPBInvACol` and `SCIPgetLPBInvCol` to retrieve, column by column, the entries of $B^{-1}A$ and B^{-1}, respectively. The columns corresponding to the basic variables will now be unit vectors. To see this, imagine reordering the columns of $\begin{bmatrix} A & I \end{bmatrix}$ so that the basic columns come first. That is, we obtain the form $\begin{bmatrix} B & N \end{bmatrix}$, where N contains the non-basic columns. Multiplying by B^{-1} gives $\begin{bmatrix} I & B^{-1}N \end{bmatrix}$. We can now move all non-basic variables to the right-hand side to obtain our desired representation of the optimization problem.

We now deal with the slightly harder task of recovering the reduced costs at the optimal tableau. The following theorem is a standard result in LP theory (see, e.g., [9]):

Theorem 3.5. Let $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, and $c \in \mathbb{R}^n$. We consider the linear program:

$$\begin{aligned} \min \quad & c^\top x \\ \text{s.t.} \quad & Ax \leq b \end{aligned}$$

which we refer to as the *primal* LP. The *dual* LP of the primal LP is given by:

$$\begin{aligned} \max \quad & y^\top b \\ \text{s.t.} \quad & A^\top y = c, \\ & y \geq 0. \end{aligned}$$

The following statements hold:

1. *Weak Duality:* If y is a feasible solution of the dual linear program, then $y^\top b \geq c^\top x$ for all feasible solutions x of the primal LP.

2. *Strong Duality:* If either the primal LP or dual LP has an optimal value, then so does the other, and their optimal values are equal.

3. *Complementary Slackness:* Let (x, y) be an optimal solution pair for the primal and dual LPs, respectively. Then, if $y_i > 0$, the i-th row of the primal LP must be tight, i.e., $\alpha_i^\top x = b_i$. Conversely, if $\alpha_i^\top x < b_i$, then $y_i = 0$.

The dual of (3.4) is

$$\begin{aligned} \max \quad & U^\top y^+ + L^\top y^- + u^\top d^+ - \ell^\top d^- \\ \text{s.t.} \quad & A^\top y^+ - A^\top y^- + d^+ - d^- = c, \\ & y^+, y^-, d^+, d^- \geq 0. \end{aligned}$$

We define $y := y^+ - y^-$ and $d := d^+ - d^-$. We will show shortly that d and $-y$ give the reduced costs for the structural and slack variables, respectively. The equality constraint reduces to

$$A^\top y + d = c.$$

Note that we have a variable y_i associated with every row of A (or every column of $A^\top$), and a reduced cost d_i for each pair of bounds (ℓ_i, u_i), i.e., for every variable x_i. SCIP provides us with the reduced cost for each structural variable via `SCIPgetColRedcost` on the corresponding `SCIP_COL` object. To get the dual solution for a row in our LP, we call `SCIProwGetDualsol` on the corresponding `SCIP_ROW` object. We have

$$c^\top x = (A^\top y + d)^\top x = d^\top x + y^\top A x = d^\top x - y^\top s.$$

Complementary slackness now implies that if x_i is basic, then it has a reduced cost of zero. On the other hand, if s_i is basic, then its associated dual solution is zero. Hence, we obtain our desired representation of the objective as a function of the non-basic variables.

3.4 Disjunction From Partial Branch-and-Bound Tree

Our algorithm requires a disjunctive set $\mathcal{D}$ such that $\mathcal{P}_I \in \bar{\mathcal{D}}$ and $\bar{x} \notin \bar{\mathcal{D}}$. We obtain this disjunction from a partial branch-and-bound tree, that is, a branch-and-bound tree with a fixed number of leaves. Other choices exist in the literature; we refer the reader to [39] for an exhaustive discussion. We have seen the branch-and-bound algorithm in Section 1.2. The key observation is that at any stage of the branch-and-bound algorithm, the set of leaves of the branch-and-bound tree forms a partitioning of the search region. Taking the union of these partitions gives us a disjunctive set $\mathcal{D}$. Since partitioning the search region by branching does not remove any integer points, we have $\mathcal{P}_I \subseteq \bar{\mathcal{D}}$.

Example 3.6. We consider the MIP

$$\begin{aligned} \max \quad & x_2 \\ \text{s.t.} \quad & -10x_1 + 4x_2 \leq -5, \\ & 2x_1 + 4x_2 \leq 13, \\ & x_1, x_2 \geq 0, \\ & x_1, x_2 \in \mathbb{Z}. \end{aligned} \tag{3.5}$$

The optimal solution to the LP relaxation is $(1.5, 2.5)$. We first branch on x_1. After adding the bound $x_1 \leq 1$, the LP relaxation gives $(1, 1.25)$. Hence, we branch on x_2. Now consider the branch $x_1 \geq 2$, where the LP solution becomes $(2, 2.25)$. We further branch on x_2. This gives four leaves corresponding to the bounds: $x_1 \leq 1 \wedge x_2 \leq 1$, $x_1 \leq 1 \wedge x_2 \geq 2$, $x_1 \geq 2 \wedge x_2 \leq 2$, and $x_1 \geq 2 \wedge x_2 \geq 3$.

An alternative disjunction would be a cross disjunction, that is, a disjunction obtained by splitting on multiple fractional variables simultaneously. In this example, the cross disjunction would consist of: $x_1 \leq 1 \wedge x_2 \leq 2$, $x_1 \leq 1 \wedge x_2 \geq 3$, $x_1 \geq 2 \wedge x_2 \leq 2$, and $x_1 \geq 2 \wedge x_2 \geq 3$. We observe that the disjunction arising from the partial branch-and-bound tree is stronger, in the sense that its disjunctive hull is a strict subset of the disjunctive hull generated by the cross disjunction.

Example 3.6 illustrates the benefit of generating a disjunction from a partial branch-and-bound tree compared to a cross disjunction. The use of partial branch-and-bound trees allows for asymmetric disjunctions (i.e., different bounds on different variables across disjunctive terms). Furthermore, each disjunctive term can be strengthened via propagation, and infeasible leaves can be pruned. Disjunctions arising from partial branch-and-bound trees are the default in our experiments. We note that since the disjunction arises from a partial branch-and-bound tree, each disjunctive term differs only in the bounds applied to the variables.

In SCIP, the creation of a partial branch-and-bound tree is done using SCIP's *probing mode.* Probing mode allows us to temporarily modify variable bounds, add constraints, and backtrack these changes. This mimics traversing a search tree. We always start at an initial node. Then we can create a new node, apply changes, create another node, and so on. Each node has a depth (e.g., the root node is at depth 0, its child at depth 1, and so forth). We can backtrack to a node of any given depth. For example, backtracking to depth 0 reverts all changes made in nodes of depth 1 and 2; backtracking to depth 1 reverts only changes made at depth 2.

On top of SCIP's probing functionalities, we implement our own algorithm to manage the branch-and-bound tree. This includes bookkeeping of explored and unexplored nodes, selecting the branching variable, and choosing the next node to explore. To achieve this, we use a binary tree data structure. Each node stores whether it is in the active search path (i.e., on the path from the root to the current node), its associated action (e.g., adding a lower bound of 5 to a variable), its parent, and its depth. We remark that each node in our implementation stores only the difference between the state at the current node and its parent—specifically, the bound changes made—thus conserving memory compared to the naive approach of storing full bounds at each node. By traversing the tree, we can reconstruct the set of bounds at any node. Algorithm 3 shows how we move the SCIP state in probing mode from one node to another. This is the same node-switching algorithm used by SCIP [38]. Roughly speaking, the idea is to first find the common ancestor of the current and target node, undo all actions from the current node to the ancestor, and then apply the actions from the ancestor to the target node. Finally, we branch on the variable with the highest pseudo-cost, and use best dual bound for node selection.

3.5 Selection of Cost Function

We now discuss the choice of objective function for the cut-generation LP. Changing the objective allows us to get a variety of cuts. The objectives we use are similar to [4] with an exception that we will explain later in this section. Similarly the proofs are expansions of those from [4]. We use primal simplex to solve (PRLP) because in the case we don't optimize to optimality e.g. because of time limit. Primal simplex returns a primal feasible solution (the algorithm sucessfully find a starting feasible basis), which is a valid cut. We initiate by checking feasibility, i.e. setting the objective to zero, we store the returned feasible basis for warm starting.

Our first objective is to find a cut as far as possible from the LP relaxation solution $\bar{x}$. Unfortunately, our normalization of $\beta = 1$ make all cuts have a violation of 1. Hence, we use as a proxy the objective function $w = e$. The interpretation of this objective is that we want to move along each ray of the basis cone from $\bar{x}$ as far as possible. In [4], a second proxy suggested is to use a new LP relaxation optimum x' obtained after first adding a single round of gomory cut. The problem is that to implement this in SCIP would require us to apply gomory separation in the middle of our separation routine. Although it is possible to implement this behavior SCIP, the solver does not offer built-in support for it. Hence, we omit this in our implementation. We also note that one could use the analytic center of the optimal face. Computing the analytic center within a plugin has been implemented in other SCIP plugins, for example in [40]. However, we also did not pursue this within the work of this thesis.

We now will introduce a class of objective functions. To motivate this class of objective functions, we first state the following results.

Theorem 3.7. Given a disjunctive set $\mathcal{D} = \cup_{i=1}^{N} \mathcal{D}_i$ and its simple relaxation $\mathcal{D}' = \cup_{i=1}^{N} \mathcal{D}_i'$ generated by some vector $c \in \mathbb{R}^n$. Then

$$\min\{c^T x \mid x \in \bar{\mathcal{D}}\} = \min\{c^T x \mid x \in \bar{\mathcal{D}}'\}$$

Algorithm 3 Node Switching

```
Input: current node $n_{\text{cur}}$, target node $n_{\text{tgt}}$.
Output: data structures updated so that the solver state equals that of $n_{\text{tgt}}$.
  1. Compute the path from $n_{\text{tgt}}$ up to the common ancestor $n_{\text{comm}}$ of $n_{\text{tgt}}$ and $n_{\text{cur}}$
     (a) $L \leftarrow$ empty list,   $q \leftarrow n_{\text{tgt}}$
     (b) While $q$ is *not* in the active search path
           i. append $q$ to $L$
          ii. $q \leftarrow \text{parent}(q)$
     (c) $n_{\text{comm}} \leftarrow q$
  2. Undo all actions on the path $n_{\text{cur}} \rightarrow n_{\text{comm}}$
     (a) $q \leftarrow n_{\text{cur}}$
     (b) While $q \neq n_{\text{comm}}$
           i. undo action($q$)
          ii. remove $q$ from active search path
         iii. $q \leftarrow \text{parent}(q)$
  3. Do all actions from $n_{\text{comm}} \rightarrow n_{\text{tgt}}$
     (a) reverse $L$
     (b) For each node $q$ in $L$
           i. apply action($q$)
          ii. add $q$ to active search path
```

Proof. Since $\bar{\mathcal{D}} \subseteq \bar{\mathcal{D}}'$, we have that the right-hand-side is less or equal than the left-hand-side. Recall the $\mathcal{V}$-polyhedral representation of the simple relaxation is given by

$$\bar{\mathcal{D}}' = \text{conv}\left(\{v^1, \dots, v^n\}\right) + \text{cone}\left(\{r_1^1, \dots, r_n^1, \dots, r_1^N, \dots, r_n^N\}\right)$$

where

$$\mathcal{D}_i' := \text{conv}(\{v^i\}) + \text{cone}(\{r_1^i, \dots, r_n^i\})$$

is the corner polyhedron obtained by solving the LP $\min\{c^T x \mid x \in \mathcal{D}_i\}$. By construction of the corner polyhedron, we have $c^T r_j^i \geq 0$ for $i \in [N]$ and $j \in [n]$. Hence,

$$\min\{c^T x \mid x \in \bar{\mathcal{D}}'\}$$

is bounded and the minimizer is attained at one of the vertices of $\bar{\mathcal{D}}'$. Since, $v_i \in \mathcal{D}_i$ for all $i \in [N]$, we have that the left-hand-side is less or equal than the right-hand-side. □

We obtain the following corollary.

Corollary 3.8. Given a disjunctive set $\mathcal{D} = \cup_{i=1}^N \mathcal{D}_i$ and its simple relaxation $\mathcal{D}' = \cup_{i=1}^N \mathcal{D}_i'$ generated by some vector $c \in \mathbb{R}^n$. There are n inequalities that can be added to the MIP such that the new LP relaxation have the LP optimal value

$$\min\{c^T x \mid x \in \bar{D}\}$$

Proof. In the proof of Theorem 3.7, we have shown that the optimal is attained at vertex v of $\bar{\mathcal{P}}'$, the n inequality can be obtained by taking any subset of n hyperplanes tight at v. □

We call the quantity $\min\{c^T x \mid x \in \bar{D}\}$ the *disjunctive lower bound.* We now want to obtain as many inequalities that is referred to in Corollary 3.8 as possible. Unfortunately, as shown in [4] not all of these inequalities can be obtained from our normalized PRLP since some of these inequalities might not necessarily cut away $\bar{x}$. We will show a corollary stating that under the condition similar to that of Theorem 2.23 at least one of these inequalities is a feasible solution to PRLP.

Corollary 3.9. Given a disjunctive set $\cup_{i=1}^N \mathcal{P}_i$ and its simple relaxation $\cup_{i=1}^N \bar{\mathcal{P}}$ generated by some vector $c \in \mathbb{R}^n$. Let $v^* \in \arg\min\{c^T v^i \mid i \in [N]\}$. If $c^T v^* > c^T \bar{x}$, then optimizing PRLP with objective cost v^*, give an optimal value of 1. In particular, it give a cut that is tight at v^*.

Proof. Clearly, the optimal value is greater or equal than 1 since $\alpha^T v^* \geq 1$ is a constraint in the normalized PRLP. Setting $\alpha = \frac{1}{c^T v^*} c$ gives the cut $\frac{1}{c^T v^*} c^T x \geq 1$ is a feasible solution to PRLP with an objective cost of 1 and is tight at v^*. □

Although theoretically a solution with an optimal value of 1 always exists, numerical issues might lead LP solvers to fail to return solutions with objective value exactly 1. In the case that an LP solver does find a solution with an objective value of 1, say α^*, we can proceed and try to find as many vertices as possible that lie on the optimal face. To do this, we first fix the optimal face by changing the constraint $\alpha^\top v^* \geq 1$ into an equality. In [4], it is suggested that we then use as objective pool the points and rays from our point-ray collection which are currently not tight at α^*. There is a natural interpretation for this: we want to find a hyperplane that is as tight as possible to the point, or, if it is a ray, we want to find a hyperplane as orthogonal as possible to the ray. We then iterate between optimizing with a point or ray as the objective and progressively discard points and rays that become tight from the objective pool. Similarly here, after changing the constraint $\alpha^\top v^* \geq 1$ into an equality, we first solve a feasibility problem to find an optimal basis. At each solve where the solver terminates with an optimal status, we update the stored basis.

We remark that in [41] an alternative method is proposed. First, we fix all variables (structural and slack) variables whose reduced cost are non-zero at α^*. Then we minimize along each variable that is not yet fixed. In our preliminary testing, these two approach almost always reduces to each other we often find that all variables in α is basic, while the reduced cost for the slack variables are given by one for the slack variable corresponding to the row $\alpha^T v^* \geq 1$ and zero elsewhere. This implies exactly changing the constraint $\alpha^\top v^* \geq 1$ into an equality. Minimizing along each slack variables with reduced cost zero is equivalent to minimizing over each objective in the suggested objective pool.

Chapter 4

Computational Result

In this chapter, we provide results on computational experiments run with our implementation of VPC in SCIP. Although our experiment design is similar to [4], we differ in that we use a single benchmark set — MIPLIB2017 [5] — rather than a collection of instances. Most importantly, we remove the working limit of 5000 rows and columns after presolving. The chapter proceeds as follows: we introduce our general setup in Section 4.1. We then proceed with root node gap closed experiments in Section 4.2. Finally, we present results from our branch-and-bound experiment in Section 4.3.

4.1 General Setup

We run all our instances on the high-performance computing cluster operated by the Zuse Institute Berlin. The cluster runs Debian 12. All runs are single-threaded on an Intel Xeon Gold 6338 (2.0 GHz). Due to resource limitations, unless otherwise stated, our runs are non-exclusive. Computing time is measured in wall-clock seconds. The time limit for each batch job is two hours and fifteen minutes, while the SCIP time limit is set to two hours. The memory limit for each batch job is set to 24 GB. The code for the experiments is publicly accessible at [42].

We implement our algorithm in Julia. The runtime environment on the cluster uses Julia version 1.11.5. We use SCIP version 9.1.0 with 8-byte precision [1], compiled without PaPILO. SCIP is built using CMake version 3.25.1, Make version 4.3, and GCC version 12.2.0, with default compile settings (`AUTOBUILD=ON`). The connection between SCIP and Julia is established via the `SCIP.jl` interface [43], where we use version 0.12.3. As the LP solver for SCIP, we use Xpress 9.6. All algorithmic communication with the LP solver is handled exclusively through SCIP's LP interface. By default, SCIP does not use Xpress presolve (`PRESOLVE`) but only quick LP presolve. However, preliminary testing shows that enabling Xpress presolve greatly benefits the solution of the PRLP, so we enable it. Unless explicitly stated otherwise, SCIP's default solver settings and random seeds are used.

Though convenient, the choice of Julia as a programming language comes with two important downsides. First, to the best of our knowledge, there is no standard method to limit Julia's memory usage. Hence, we can only detect that the memory limit is being hit when the job is killed by the `SLURM` system. We indicate when this occurs in the upcoming sections and describe the corresponding remedial actions taken. Second, Julia is a just-in-time compiled language, meaning code is compiled when it is run, with caching being used such that code is not recompiled after its first run unless the code is changed. To minimize the variability of just-in-time compilation, we first do a single run of the algorithm on the instance `neos5` before the batch experiment is run. The instance is selected because it goes through all of the expensive subroutines and ensures that these subroutines are compiled and cached before we run the batch experiment.

As our testbed, we use MIPLIB 2017 [5]. Statistics for MIPLIB 2017 problems are shown in

Appendix A. We discard certain instances to conserve computational resources. The discarded instances, along with the reasoning for their exclusion, are provided in the description of each experiment. On all runs, SCIP default presolving is used.

Unless otherwise stated, the parameters used to run the VPC separation routine are as follows: We allocate a time limit of 900 seconds for each of the following steps in the VPC process: generating a partial branch-and-bound tree with 64 leaves, constructing the PRLP, and solving the PRLP with different objectives. The LP solver is given 120 seconds to find an initial feasible solution to the PRLP and another 120 seconds to find a feasible solution to the PRLP after the tightening of the constraint $\alpha^\top v^* \geq 1$ described in Section 3.5. The time limit for all other objectives is 10 seconds. The PRLP solving process is terminated if failures occur for 10 different objective functions in a row. Finally, when the shifted geometric mean is reported, we use a shift of 1.

4.2 Root Node Gap Closed

A standard measure of evaluating the quality of cutting plane generation algorithms is by examining the improvement of the gap between the LP relaxation and a known optimal integer solution before and after adding the cuts, that is the quantity,

$$\% \text{ root node gap closed} = 100 \times \frac{c^\top x' - c^\top \bar{x}}{c^\top x_I - c^\top \bar{x}},$$

where x' is an optimum solution to the LP relaxation after adding cuts, $\bar{x}$ is an optimum solution to the LP relaxation before adding cut and x_I is a known optimal integer solution. This was done in [4], [16], and [44]. There is two reasons why we look at this metric: first it is more optimistic since the addition of cutting plane may affect various aspect of the branch-and-bound process it is generally difficult to specify reasons for performance improvement or degradation. Root node gap as a metric isolates the performance of the cut generation algorithm from other parts of the solver. Second, the metric is still generally indicative of the quality of the cutting plane algorithm while requiring relatively low computational resource in comparison to full branch-and-bound experiments allowing us to explore different parameterization for our cuts. Though this is a good heuristic, we note that tighter LP relaxation does not always translate to smaller branch-and-bound tree and hence better branch-and-bound performances (e.g. see [45]). It also couldn't be applied to feasibility instance, although in such instances cutting plane could still be very useful to improve the integrality of solutions.

For all experiments in this set, root node propagation is disabled along with heuristic by setting SCIP heuristic emphasis to off. We exclude the following instances from the analysis:

1. `supportcase19` and `highschool1-aigio`, because the first LP solve exceeded our time limit of two hours.

2. `rd-rplusc-21`, `neos-1122047`, and `leo2` are excluded because the first dual bound reported at the root node originates from sources other than the initial LP solve, such as presolving. To identify such cases, we compute

$$\mu = \frac{|\text{first\_lp\_value} - \text{first\_reported\_dual\_bound}|}{|\max(1, \text{first\_reported\_dual\_bound})|},$$

   and discard instances with $\mu \geq 0.01$. The motivation for this exclusion is that the dual bound obtained from the LP relaxation may be weaker than the one produced by presolving. In such cases, even after adding cuts, the final bound might still be worse than the original presolve bound, rendering the gap-closure metric meaningless—since the initial and final dual bounds do not both originate from LP solves and therefore do not serve as a valid measure of how much the LP relaxation has been tightened.

3. `bnatt500`, `cryptanalysiskb128n5obj14`, `cryptanalysiskb128n5obj16`, `ns1116954`, `neos859080`, `fhnw-binpack4-4`, `fhnw-binpack4-48`, `gfd-schedulen180f7d50m30k18`, `neos-2075418-temuka`, `neos-3004026-krka`, `neos-3402454-bohle`, `neos-3988577-wolgan`, and `ns1952667`, because they are tagged with `feasibility` or `infeasible`. As we pointed

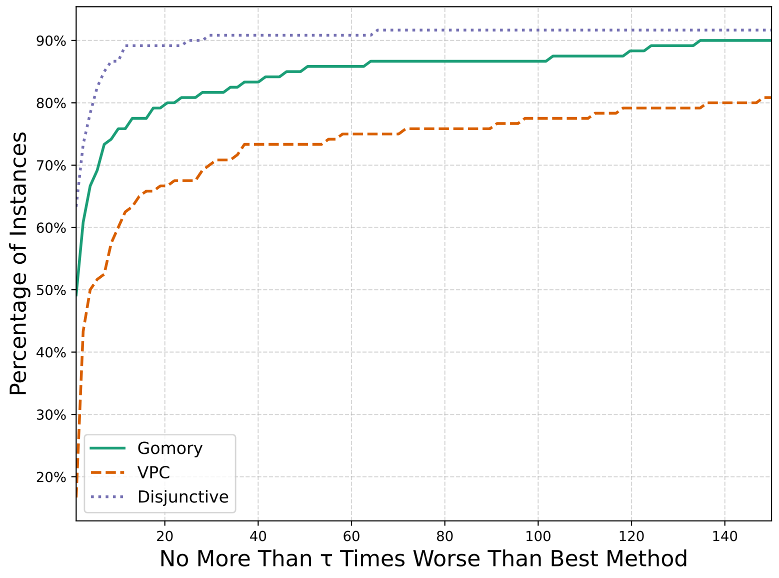

Figure 4.1: Performance profile of the root node gap closed after a single round of Gomory cuts versus a single round of VPCs, restricted to instances where at least one VPC was generated.

before, for this instances the metric is not applicable since we are only finding a feasible solution (in the feasibility case) and the objective function doesn't play a role. In the infeasibility case, the metric is not applicable since we do not have a reference point x_I.

We define `Set A` to be the set all instances that are not excluded under any of the 3 criterias above. We define `Set B` to be the set of all instances of `Set A` that additionally satisfy

4. Running VPC separation routine after solving the first root node LP relaxation yields at least 1 cut.

A list of `Set B` instances is given in Appendix C. The full result of the experiment in this section is given in Appendix B.

We begin by comparing the root node gap closed after applying one round of VPCs the root node gap closed after applying one round of Gomory cuts, following [4]. Since SCIP does not provide a pure Gomory implementation, rather one where CMIR function is applied to the aggregated row, we use the version included in the example code accompanying SCIP [46] to match the design in [4]. We remove the limit on the number of supporting variables per cut, disable the separation of rows with integral slack, and force the addition of generated cuts to the LP by setting `forcecut=true` in the `SCIPaddRow` method. The number of cuts generated is limited to $\min(2000, p)$, where p is the number of fractional variables in the first LP solution. All other settings remain at their default values. We do our analysis over `Set B`.

In Figure 4.1, we present a performance profile of the root node gap closed after one round of Gomory cuts, one round of VPCs, and the disjunctive lower bounds. We observe that VPCs underperform compared to Gomory cuts, while the gap closed at the disjunctive lower bound exceeds that of the Gomory cuts. This trend is further reflected in the shifted geometric means of the root node gap closed by Gomory cuts, VPCs, and the disjunctive lower bound, which are 4.18%, 2.29%, and 5.71%, respectively. This is noteworthy, since the disjunctive lower bound is theoretically attainable by VPCs. This observation is consistent with the findings reported in [4], based on a different but partially overlapping set of instances, where over their instance set they also reported that in the majority of instances gomory outperforms VPC. It has been hypothesized in [47] that the performance gap between VPCs and Gomory cuts could be reduced through monoidal strengthening of VPCs. However, this lies beyond the scope of the present thesis.

We now examine individual instances. We declare a VPC win if VPCs close at least one percentage point more root node gap than Gomory cuts, a Gomory win if Gomory cuts close at

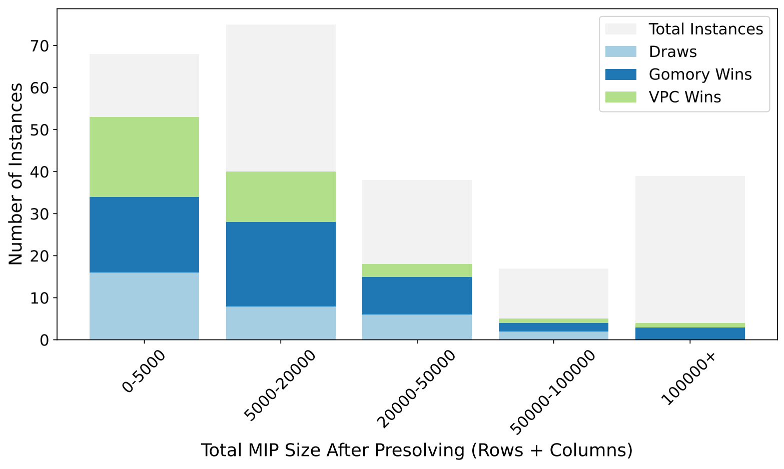

Figure 4.2: Stacked bar chart showing the number of instances where Gomory cuts win, VPCs win, or the result is a draw, grouped by MIP size after presolving (sum of rows and columns). The total number of `Set A` instances in each bin is indicated by the gray background bars, illustrating the proportion of instances in which VPCs are generated.

least one percentage point more than VPCs, and a draw otherwise. There are a total of 52 Gomory wins, 36 VPC wins, and 32 draws. On 37 and 14 instances, respectively, Gomory cuts and VPCs win by more than 10 percentage points.

Figure 4.2 shows a stacked bar chart where the vertical axis indicates the number of instances where Gomory cuts win, VPCs win, or the result is a draw. The bars are grouped by MIP size after presolving (sum of rows and columns). We observe that as the size of the MIP increases, the proportion of instances where VPCs win against Gomory cuts tends to decrease. During private correspondence with the author of [4], it was hypothesized that this is due to decreasing numerical stability of the LP tableau as the MIP size increases. Unfortunately, standard metrics for measuring stability of LP tableau such as MIP kappa is not yet implemented in SCIP. Regardless, this merits further investigation, although is currently beyond the scope of the thesis. Also noteworthy is the proportion of instances in which VPCs are successfully generated in comparison to the total number of `Set A` instances in each bin, shown in the gray background bars. We observe that the likelihood significantly degrades once the MIP size exceeds 5000 rows and columns. In terms of comparison with [4] this is significant as there a strict limit on the dimensionality of the MIP is given.

In contrast, Figure 4.3 shows a stacked bar chart that groups instances by the composition of variables: binary and continuous versus general integer, binary, and continuous. We observe that the proportion of VPC wins is significantly higher among instances consisting only of binary and continuous variables. The observation that VPC tends to be more effective on binary instances is also reported in [4]. Our hypothesis is that propagation has a significantly stronger effect on instances consisting solely of binary and continuous variables compared to those that also include general integer variables. Contrary to the trend observed with MIP size, we did not observe a significant difference in the likelihood of successful VPC generation between the two categories.

Finally, we consider the reasons for failure to generate VPCs. These are summarized in Table 4.1. We observe that failure to prove feasibility for PRLPs or tightened PRLPs account for the largest proportion of failures. As discussed earlier, this may be due to numerical issues or the optimal LP relaxation residing inside the relaxed disjunctive hull.

We did not generate VPCs when some free non-basic variables were fixed to zero, as indicated by the basis status `SCIP_BASESTAT_ZERO` in SCIP. Another cause of failure involves problems with large LP relaxations and dense tableaus. These may cause our algorithm to hit the time limit while collecting the points and rays necessary for the PRLP, or to hit the memory limit.

From a practical perspective, this last failure type can be mitigated by imposing strict limits on the size of the MIP instances for which our separator is invoked.

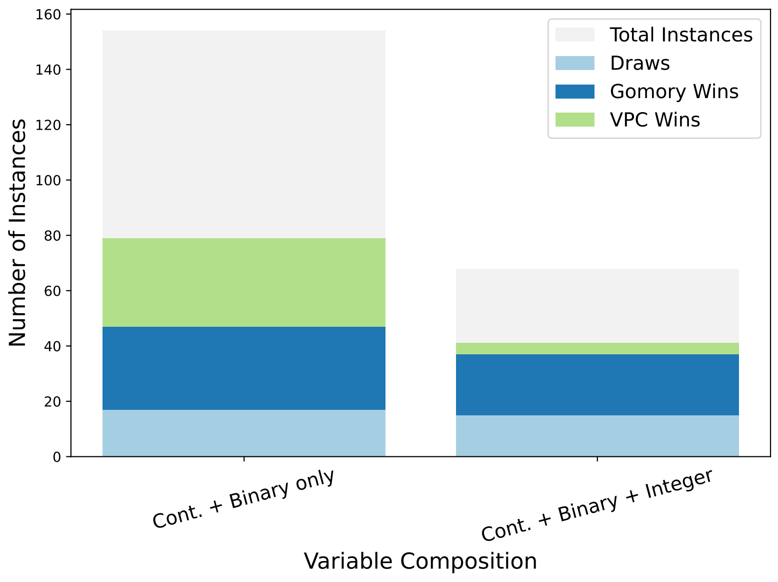

Figure 4.3: Stacked bar chart showing the number of instances where Gomory cuts win, VPCs win, or the result is a draw, grouped by variable composition (binary and continuous vs. general integer, binary, and continuous). The gray background bars indicate the total number of `Set A` instances in each bin.

Failure Reason	Number of Instances
Failure to prove PRLP or tightened PRLP feasibility	49
Hit time limit while generating the partial branch-and-bound tree	16
Hit time limit while collecting points and rays	15
Encountered non-basic free variable fixed to zero	10
LP error during partial branch-and-bound tree generation	5
VPC not run	4
Memory limit	3

Table 4.1: Summary of reasons for failure to generate VPCs.

We now consider the performance of VPCs in conjunction with SCIP's default cuts. This is an important and more realistic simulation of the scenario where solvers applies a variety of cuts. Specifically, we compare three strategies: using only SCIP's default cuts, applying a single round of VPCs before SCIP's default cuts, and applying SCIP's default cuts first followed by a single round of VPCs. Our analysis is restricted to the 120 instances in which at least one VPC was generated in previous experiments. To apply VPCs before other SCIP cuts, we assign the separator a priority of 99999. To apply VPCs after SCIP's cuts, we assign the separator a priority of -99999 and set `delayed=true`. The time limit constraint for VPC generation is maintained. For improved comparability, we cap the number of VPCs generated per round at 200. Additionally, instead of adding the generated VPCs directly to the LP, we insert them into SCIP's global cut pool, allowing SCIP to manage cut selection. During the run in which SCIP's default cuts are applied first, followed by a round of VPCs, the memory limit is exceeded for instances `roi5alpha10n8`, `nexp-150-20-8-5`, and `proteindesign121hz512p9`. In these cases, we copy the gap closed by SCIP's default cuts to fill in the missing values.

In 119 out of 120 instances, a VPC was successfully generated when applied before SCIP's default cuts. The only instance in which VPC generation failed was `rmatr100_p10`due to the PRLP feasibility check, which approached the time limit in an earlier experiment but exceeded it in the current run. However, in only 74 out of 120 instances was a VPC generated after applying SCIP's default cuts. The reasons for this failure are summarized in Table 4.2.

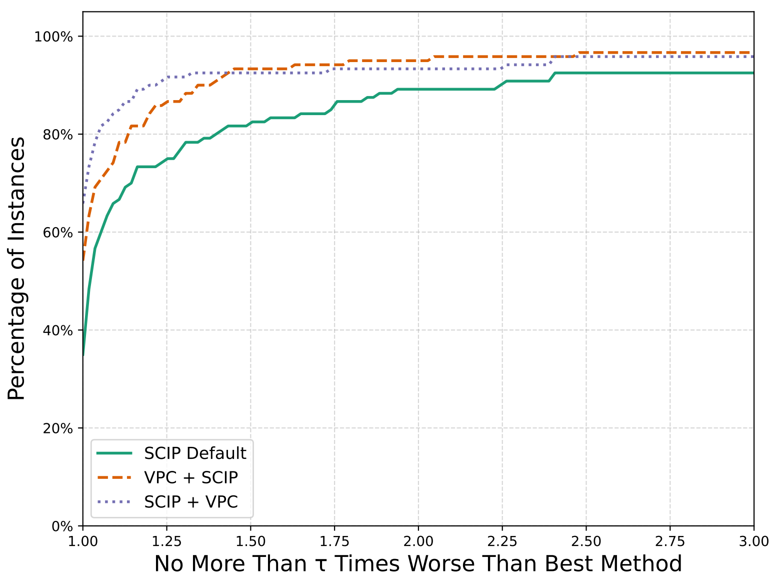

Figure 4.4: Performance profile of the root node gap closed at the end of root node processing, comparing configurations where no VPC is applied, VPC is applied before the separation loop, and VPC is applied after the separation loop.

Figure 4.4 shows the performance profile of the three strategies. We observe that VPCs enhance the performance of SCIP's default cutting planes. This is further reflected in the shifted geometric means of the root node gap closed, which are 14.60%, 17.10%, and 17.60% for SCIP default cuts, VPCs applied before SCIP cuts, and VPCs applied after SCIP cuts, respectively. Notably, this observation is consistent with findings in [4], where applying VPCs at the end of a solver's default separation loop on a different solver also led to improvements in root node gap closed. Given that the geometric mean is highest when VPCs are applied after SCIP cuts, we focus exclusively on this configuration in the subsequent analysis.

Failure Reason	Number of Instances
Failure to prove PRLP or tightened PRLP feasibility	33
Hit time limit while generating the partial branch-and-bound tree	1
Hit time limit while collecting points and rays	0
Encountered non-basic free variable fixed to zero	4
LP error during partial branch-and-bound tree generation	4
VPC not run	1
Memory limit	3

Table 4.2: Summary of reasons for failure to generate VPCs after SCIP default cuts over the `Set B` instances.

A per-instance breakdown over instances where at least one VPC is generated in either VPC configuration reveals that one of the VPC configurations improves performance by at least 1% over the SCIP default on 54 instances. In 22 of these, both VPC configurations outperform the SCIP default. In 16 instances, only applying VPC before SCIP cuts results in an improvement, whereas in the other 16 instances, only applying VPC after SCIP cuts leads to better performance.

Conversely, in 13 instances, adding VPCs before SCIP's default cuts degrades the root node gap closed. This is not unexpected, as early cut insertion can significantly alter the solver's trajectory. More surprisingly, in 5 instances, enabling VPCs after SCIP's default cuts leads to degraded performance—even though no VPCs are generated. This is noteworthy, as we would expect that enabling VPCs post-SCIP should not affect the composition of cutting planes generated by SCIP.

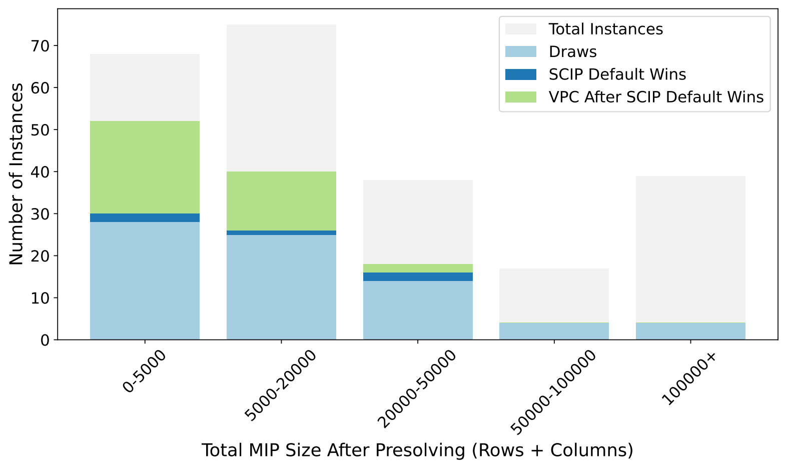

Figure 4.5: Stacked bar chart showing the number of instances where SCIP default cuts win, applying VPC after SCIP default cuts win, or the result is a draw, grouped by MIP size after presolving (sum of rows and columns). The total number of MIPLIB2017 instances in each bin is indicated by the gray background bars.

However, we observed that the branch-and-bound search path diverges when VPCs are scheduled after SCIP cuts, implemented by assigning a lower priority to the VPC separator. This is indicated in the solver logs, where the number of calls to other separators is also affected. In no instance where VPCs are actually generated does enabling VPCs after SCIP cuts degrade performance.

As before, in Figure 4.5 we plot a stacked bar chart, this time comparing SCIP default cuts against applying VPC after SCIP default cuts. We only mark a VPC win if at least one VPC cut is found. As expected, a large number of instances result in a draw, since VPCs do not enhance the performance of SCIP default cuts in those cases. However, we again observe that the benefit of VPC is more likely to occur when applied to smaller instances.

In Figure 4.6, we plot a stacked bar chart, where instances are binned by variable composition (binary and continuous only versus general integer, binary, and continuous). We still see that VPC performs better on instance without any general integers. However, in the bin with general integers, VPC performance improves: in the single-cut experiment, VPC outperformed gomory mixed-integer cuts in only 4 out of 41 instances, whereas when applied after SCIP default cuts, it does so in 9 out of 41 instances. Our hypothesis is that in the integer case, VPC benefits from working on a formulation already strengthened by earlier cuts, making it more effective. This merits further investigation.

We conclude that although VPC alone closes less root node gap than Gomory cuts, using VPC in conjunction with `SCIP` default cuts may still be beneficial. We also note that, since applying VPC after SCIP default cuts yields superior results, this configuration will be used as the default setting for the subsequent experiments.

We now describe a slightly separate experiment designed to test the practical implications of Theorem 2.18. Our hypothesis is that, since extreme rays are facet-defining, solving methods for PRLP that return a basic feasible solution should yield stronger cuts than those that do not. A simple way to test this is to solve the PRLP using the barrier method while varying whether or not crossover is applied. We limit the number of cuts generated to the number of fractional variables.

The results show that the barrier method with crossover is clearly superior. On 116 instances, both configurations generated more than one cut. The shifted geometric means of the root node gap closed across these instances are 0.51% and 0.19% for barrier with and without crossover, respectively. On 36 instances, barrier with crossover closed more than 1% additional gap compared to barrier without crossover, with over 10% more gap closed on four of them. Only on one instance—`var-smallemery-m6j6`—did barrier without crossover close more than 1% additional gap, by around 2%.

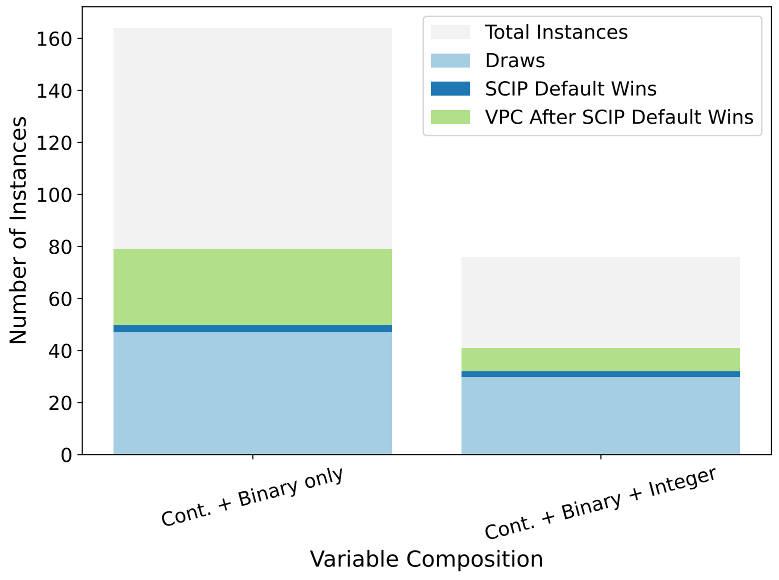

Figure 4.6: Stacked bar chart showing the number of instances where SCIP default cuts win, applying VPC after SCIP default cuts win, or the result is a draw, grouped by the composition of variables (binary and continuous vs. general integer, binary, and continuous). The total number of MIPLIB2017 instances in each bin is indicated by the gray background bars.

Table 4.3: Number of Instances with VPCs Generated

	VPCs generated in at least one run	VPCs generated in all three runs
No Restart	80	52
One Restart	56	42

4.3 Branch-and-Bound Performance

We now evaluate whether or not VPCs improve performance when integrated into a branch-and-bound solver. Since VPC is computationally expensive in comparison to other cutting planes, we perform only one round at the root node. It is typical in MIP solvers for more expensive cuts to be activated only after a restart in the branch-and-bound process. A discussion of restart strategies in MIP can be found in [48]. For this reason, we compare three configurations in this chapter: VPCs not activated (`default` configuration), VPCs activated without any restart requirement (`norestart` configuration), and VPCs activated only after SCIP restarts the branch-and-bound process at least once (`onerestart` configuration). We apply VPCs after the default SCIP cuts at the root node and keep the time limit from the previous experiments. To compensate for solver variability, we permute each instance with three different random seeds (25, 644, and 856) using the `SCIPpermuteProb` function. All our runs are also *proof-of-optimality* runs; we supply SCIP with the reference solutions provided by the MIPLIB2017 benchmark set. The solve time and node count reported are the shifted geometric mean from the three runs. We note that we obtain the random seeds from a random number generator. The full results for the experiments described here are given in Appendix D.

Table 4.3 shows the number of instances in which VPCs were generated under the various settings. First, we see that the set of instances where VPCs are applicable constitutes only a minority of the MIPLIB 2017 instances. Hence, enabling them by default is likely to result in unnecessary computational effort. We also see that the permutation of instances affects whether or not a VPC can be generated. This is expected, as permuting an instance may change the LP solution from which we cut, the disjunction that is obtained, and consequently also the PRLP that is formulated.

We now investigate whether, in instances where VPCs are generated, they are beneficial for the branch-and-bound process. We consider three metrics: the number of instances solved to optimality, solve time, and node count.

We begin by examining the number of instances solved to optimality and their corresponding solve times. Comparing the `default` and `norestart` configurations, there are 52 instances in which VPCs are generated by `norestart` in all three runs. Among these, `norestart` solves 27 instances to optimality in all three runs, while `default` solves 31. There are no instances that `norestart` solves in all three runs which are not also solved by `default`. The shifted geometric mean of solve times across the 27 instances solved by both configurations is 309.96 s for `default` and 773.55 s for `norestart`. More notably, even when the cuts are provided for free—that is, when the time required to generate the cuts is excluded—`norestart` remains inferior, with a shifted geometric mean of 369.03 s compared to 309.96 s for `default`. When looking at individual instances, `norestart` is faster than `default` in only two instances, and in four additional instances if the cut generation time is not counted.

A similar trend is observed when comparing `default` and `onerestart`. VPCs are generated in all three runs in 42 instances by `onerestart`. Of these, `default` solves 21 instances to optimality in all three runs, compared to 20 solved by `onerestart`. Again, the set of instances solved by `onerestart` is a strict subset of those solved by `default`. The shifted geometric mean of solve times across these 20 instances is 522.46 s for `default` and 1054.63 s for `onerestart`. Even when cuts are given for free, `onerestart` still performs worse, with a shifted geometric mean of 607.45 s. When looking at individual instances, `onerestart` is not faster than `default` in any instance and is faster in only five instances if the cut generation time is excluded.

Hence, using both the number of instances solved to optimality and solve time as performance metrics, the application of VPCs in either alternative configuration results in inferior performance compared to `default`.

We make a brief note regarding the comparison between the `default` and `norestart` configurations with respect to solve time. The two configurations both generate at least one cut and solve to optimality in all three runs on a total of 16 instances. The shifted geometric mean of solve times across these instances is 886.41 s for `default` and 939.44 s for `norestart`. Hence, on the instances we tested, applying VPCs only after a restart appears to be less effective than applying them before a restart.

We next consider node count, which offers a complementary perspective on the impact of VPCs. For the node count comparison between `default` and `norestart`, we again consider the 27 instances in which both configurations solve the instance to optimality in all three runs and where `norestart` generates at least one VPC. The shifted geometric means of node count across these instances are 3355.74 for `default` and 3351.78 for `norestart`. However, in the `norestart` configuration, nodes are solved more slowly. The shifted geometric mean of time per node is 1.23 s for `norestart`, compared to 0.48 s for `default`. There are six instances in which `norestart` uses at least 10% fewer nodes than `default`; all six involve only binary and continuous variables and have a presolved MIP size below 50,000 rows and columns. Conversely, in nine instances, `default` uses at least 10% fewer nodes than `norestart`. We note that many of these instances also consist only of binary and continuous variables and have a presolved MIP size below 50,000. Hence, these structural properties alone do not appear sufficient for VPCs to be effective.

Next, we consider the comparison between `default` and `onerestart`. There are 20 instances in which the `onerestart` configuration generates more than one VPC in all three runs, and both configurations solve the instance to optimality in all three runs. The shifted geometric means of node count across these instances are 10091.14 for `default` and 9751.29 for `onerestart`. As in the case of `norestart`, enabling VPCs in the `onerestart` configuration increases the time per node. The shifted geometric means of time per node are 0.27 s for `default` and 0.85 s for `onerestart`. On three instances, `onerestart` achieves a node count at least 10% lower than that of `default`.

Finally, we compare `norestart` and `onerestart`. There are 16 instances where both configurations solve the instance to optimality and generate at least one VPC in all three runs. `norestart` achieves a lower shifted geometric mean of node count compared to `onerestart`, with 4008.67 and 4511.13, respectively. However, `onerestart` achieves a slightly better shifted geometric mean of time per node, with 1.1448 s compared to 1.2541 s for `norestart`. On individual instances, `onerestart` is at least 10% better than `norestart` in one instance, while `norestart` outperforms `onerestart` by at least 10% in six instances.

Hence, we conclude that applying VPCs leads to a trade-off between node count and time per node. Although both VPC configurations yield lower branch-and-bound node counts compared to `default`, they require longer solve times overall.

Conclusion

We begin this thesis with the question of whether the result of [4] can be reproduced within the context of a different solver and over a single benchmark set instead of a collection of instances. At the end of this thesis, the answer seems to be yes, partially. Before we elaborate on the answer, we will first summarize what we have done throughout this thesis.

In this thesis, we have reviewed the theoretical background that was established in [4]. We have expanded details of the proofs that are originally only sketched in the paper. We have also extended the theory by proposing that, rather than optimizing over each disjunctive set with the cost vector of the MIP to obtain the corner polyhedron relaxation, one could instead opt to use a cost vector for which the LP solution is the unique minimizer. Polyhedral theory tells us that such a direction always exists. We also established that when one uses such a direction instead, the auxiliary LP is always guaranteed to be feasible. To emphasize the contrast, we also show examples where such a guarantee does not exist in general for the original scheme proposed in [4].

Next, we explore the intricacies of implementing a theoretical cutting plane within a branch-and-bound solver. We discuss details on how the input of the $\mathcal{V}$-polyhedral cut separation LP—the corner polyhedron relaxation information—can be obtained efficiently from the LP solver. In the process, we also explain how SCIP provides access to LP information via the LPI interface. We also discuss how one can implement a partial branch-and-bound algorithm within SCIP's probing mode. Finally, we discuss the choice of objective function for the auxiliary LP. We highlight the connection between the scheme proposed in [4] and other schemes in the literature for obtaining multiple vertices from the optimal face, such as [41].

Finally, we provide computational experiments based on our implementation of VPC in SCIP. As our test set, we use the full MIPLIB2017 instance set [5], in contrast to the partial subset used in [4]. We found that VPC can only be applied to a subset of the MIPLIB2017 instances, which we label `Set B` instances. We then perform extensive analysis on the root node gap closed and the branch-and-bound effect of VPC over this instance set. In our root node gap closed experiments, we reinforce two key findings of [4]: First, although in general the cut given by VPC alone is weaker than Gomory mixed-integer cuts, VPC enhances the performance of the solver's default cuts. Second, we also show that VPC has a tendency to perform better on instances where all integer variables are additionally binary constrained. We also observe that VPC tends to perform better on instances with smaller dimensions. This is a new insight, as in [4] small instance sizes of less than 5000 rows and columns were used as an instance selection criterion. Hence, the performance of VPC on larger instances has never been reported.

We obtain a less positive result in our branch-and-bound experiment. We are unable to identify a scenario where VPC improves the default performance of the solver. We remark that the finding that VPC in general degrades solver performance is also reported in [4]. Hence, the question of identifying instances amenable to VPCs remains open. In addition, we report the even stronger observation that even if VPCs are given for free, in general, adding VPC still does not improve solver performance. Finally, we note that although VPC does not improve solve time in general, it does tend to reduce node count. An explanation for this apparent discrepancy is found when evaluating time per node as a metric. We observed that the time spent on each node increases despite the node count going down, which explains why a lower node count does not translate to faster time to optimality.

Bibliography

[1] S. Bolusani, M. Besançon, K. Bestuzheva, A. Chmiela, J. Dionísio, T. Donkiewicz, J. van Doornmalen, L. Eifler, M. Ghannam, A. Gleixner *et al.*, "The scip optimization suite 9.0," *arXiv preprint arXiv:2402.17702*, 2024.

[2] FICO, "Xpress Optimizer v9.6 reference manual," https://www.fico.com/fico-xpress-optimization/docs/latest/overview.html, 2025, accessed: 2025-06-27.

[3] T. Achterberg and R. Wunderling, "Mixed integer programming: Analyzing 12 years of progress," in *Facets of combinatorial optimization: Festschrift for martin grötschel.* Springer, 2013, pp. 449–481.

[4] E. Balas and A. M. Kazachkov, "V-polyhedral disjunctive cuts," *arXiv preprint arXiv:2207.13619*, 2022.

[5] A. Gleixner, G. Hendel, G. Gamrath, T. Achterberg, M. Bastubbe, T. Berthold, P. Christophel, K. Jarck, T. Koch, J. Linderoth *et al.*, "Miplib 2017: data-driven compilation of the 6th mixed-integer programming library," *Mathematical Programming Computation*, vol. 13, no. 3, pp. 443–490, 2021.

[6] J. Forrest, T. Ralphs, S. Vigerske, H. G. Santos, J. Forrest, L. Hafer, B. Kristjansson, jpfasano, EdwinStraver, Jan-Willem, M. Lubin, rlougee, a andre, jpgoncal1, S. Brito, h-i gassmann, Cristina, M. Saltzman, tosttost, B. Pitrus, F. MATSUSHIMA, P. Vossler, R. . SWGY, and to st, "coin-or/cbc: Release releases/2.10.12," Aug. 2024. [Online]. Available: https://doi.org/10.5281/zenodo.13347261

[7] M. Miltenberger, *Linear Programming in MILP Solving-A Computational Perspective.* Verlag Dr. Hut GmbH, 2023.

[8] Gurobi Optimization, LLC, "Gurobi Optimizer Reference Manual," 2024. [Online]. Available: https://www.gurobi.com

[9] R. J. Vanderbei, *Linear Programming: Foundations and Extensions.* Springer Science & Business Media, 2013, vol. 196.

[10] A. Schrijver, *Theory of linear and integer programming.* John Wiley & Sons, 1998.

[11] L. A. Wolsey, *Integer programming.* John Wiley & Sons, 2020.

[12] M. Conforti, G. Cornuéjols, G. Zambelli, M. Conforti, G. Cornuéjols, and G. Zambelli, *Integer programming models.* Springer, 2014.

[13] T. Hürlimann, *Puzzles and games: a mathematical modeling approach.* Lulu. com, 2016.

[14] A. Land and A. Doig, "An automatic method of solving discrete programming problems," *Econometrica: Journal of the Econometric Society*, pp. 497–520, 1960.

[15] E. Balas, "Intersection cuts—a new type of cutting planes for integer programming," *Operations Research*, vol. 19, no. 1, pp. 19–39, 1971.

[16] E. Balas, S. Ceria, G. Cornuéjols, and N. Natraj, "Gomory cuts revisited," *Operations Research Letters*, vol. 19, no. 1, pp. 1–9, 1996.

[17] K. Andersen, Q. Louveaux, R. Weismantel, and L. Wolsey, "Cutting planes from two rows of a simplex tableau," *Proceedings of IPCO XII*, vol. 4513, pp. 1–15, 2007.

[18] Q. Huangfu and J. J. Hall, "Parallelizing the dual revised simplex method," *Mathematical Programming Computation*, vol. 10, no. 1, pp. 119–142, 2018.

[19] A. Koberstein, "The dual simplex method, techniques for a fast and stable implementation," Ph.D. dissertation, Paderborn University, Germany, 2005.

[20] R. Wunderling, "Paralleler und objektorientierter simplex," Ph.D. dissertation, 1996.

[21] R. E. Bixby, "Solving real-world linear programs: A decade and more of progress," *Operations research*, vol. 50, no. 1, pp. 3–15, 2002.

[22] G. M. Ziegler, *Lectures on polytopes.* Springer Science & Business Media, 2012, vol. 152.

[23] M. Joswig and T. Theobald, *Polyhedral and algebraic methods in computational geometry.* Springer Science & Business Media, 2013.

[24] J. Edmonds, "Maximum matching and a polyhedron with 0, 1-vertices," *Journal of research of the National Bureau of Standards B*, vol. 69, no. 125-130, pp. 55–56, 1965.

[25] R. R. Meyer, "On the existence of optimal solutions to integer and mixed-integer programming problems," *Mathematical Programming*, vol. 7, pp. 223–235, 1974.

[26] R. E. Gomory, "An algorithm for the mixed integer problem," *Report No. P-1885, The Rand Corporation, Santa Monica, CA.*, 1960.

[27] W. Cook, S. Dash, R. Fukasawa, and M. Goycoolea, "Numerically safe gomory mixed-integer cuts," *INFORMS Journal on Computing*, vol. 21, no. 4, pp. 641–649, 2009.

[28] G. Cornuéjols, "Valid inequalities for mixed integer linear programs," *Mathematical programming*, vol. 112, no. 1, pp. 3–44, 2008.

[29] H. Marchand, A. Martin, R. Weismantel, and L. Wolsey, "Cutting planes in integer and mixed integer programming," *Discrete Applied Mathematics*, vol. 123, no. 1-3, pp. 397–446, 2002.

[30] M. Turner, T. Koch, F. Serrano, and M. Winkler, "Adaptive cut selection in mixed-integer linear programming," *arXiv preprint arXiv:2202.10962*, 2022.

[31] S. S. Dey and M. Molinaro, "Theoretical challenges towards cutting-plane selection," *Mathematical Programming*, vol. 170, pp. 237–266, 2018.

[32] F. Wesselmann and U. Stuhl, "Implementing cutting plane management and selection techniques," in *Technical Report.* University of Paderborn, 2012.

[33] E. Balas, "Disjunctive programming," *Annals of discrete mathematics*, vol. 5, pp. 3–51, 1979.

[34] E. Balas and R. G. Jeroslow, "Strengthening cuts for mixed integer programs," *European Journal of Operational Research*, vol. 4, no. 4, pp. 224–234, 1980.

[35] E. Balas and F. Margot, "Generalized intersection cuts and a new cut generating paradigm," *Mathematical Programming*, vol. 137, no. 1, pp. 19–35, 2013.

[36] M. Perregaard and E. Balas, "Generating cuts from multiple-term disjunctions," in *International conference on integer programming and combinatorial optimization.* Springer, 2001, pp. 348–360.

[37] Q. Louveaux, L. Poirrier, and D. Salvagnin, "The strength of multi-row models," *Mathematical Programming Computation*, vol. 7, no. 2, pp. 113–148, 2015.

[38] T. Achterberg, "Constraint integer programming," Ph.D. dissertation, 2007.

[39] A. M. Kazachkov, "Non-recursive cut generation," Ph.D. dissertation, Ph. D. thesis, Carnegie Mellon University, 2018.

[40] M. Turner, T. Berthold, M. Besançon, and T. Koch, "Cutting plane selection with analytic centers and multiregression," in *International Conference on Integration of Constraint Programming, Artificial Intelligence, and Operations Research.* Springer, 2023, pp. 52–68.

[41] T. Berthold and D. Salvagnin, "Cloud branching," in *Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems: 10th International Conference, CPAIOR 2013, Yorktown Heights, NY, USA, May 18-22, 2013. Proceedings 10.* Springer, 2013, pp. 28–43.

[42] G. Tjusila, "Vpolycut," https://github.com/gtjusila/VPolyCut.

[43] SCIP.jl contributors, "Scip.jl: Julia interface to the scip optimization suite," https://github.com/scipopt/SCIP.jl.

[44] P. Bonami, "On optimizing over lift-and-project closures," *Mathematical Programming Computation*, vol. 4, pp. 151–179, 2012.

[45] P. Shah, S. S. Dey, and M. Molinaro, "Non-monotonicity of branching rules with respect to linear relaxations," *INFORMS Journal on Computing*, 2025.

[46] SCIP Optimization Suite, "Gomory mixed integer cuts in SCIP," https://www.scipopt.org/doc/html/GMI_MAIN.php, 2024, accessed: 2025-06-07.

[47] A. M. Kazachkov and E. Balas, "Monoidal strengthening of simple v-polyhedral disjunctive cuts," *Mathematical Programming*, pp. 1–24, 2025.

[48] G. C. Hendel, "Adaptive solver behavior in mixed-integer programming/adaptives löserverhalten für gemischt ganzzahlige programmierung," 2022.

Chapter A

Statistics on the MIPLIB 2017 instance set

Table A.1: Statistics for the MIPLIB 2017 instance set. Columns from left to right are: Instance name, number of variables (#Var), binary variables (#Bin), general integers (#Int), continuous variables (#Cont), constraints (#Cons), objective value (Obj), and a marker (F) indicating whether the instance is a feasibility or infeasible instance.

Instance	#Var	#Bin	#Int	#Cont	#Cons	Obj	F
30n20b8	18380	18318	62	0	576	302.00	-
50v-10	2013	1464	183	366	233	3311.18	-
academictimetablesmall	28926	28926	0	0	23294	0.00	-
air05	7195	7195	0	0	426	26374.00	-
app1-1	2480	1225	0	1255	4926	-3.00	-
app1-2	26871	13300	0	13571	53467	-41.00	-
assign1-5-8	156	130	0	26	161	212.00	-
atlanta-ip	48738	46667	106	1965	21732	90.01	-
b1c1s1	3872	288	0	3584	3904	24544.25	-
bab2	147912	147912	0	0	17245	-357544.31	-
bab6	114240	114240	0	0	29904	-284248.23	-
beasleyC3	2500	1250	0	1250	1750	754.00	-
binkar10_1	2298	170	0	2128	1026	6742.20	-
blp-ar98	16021	15806	0	215	1128	6205.21	-
blp-ic98	13640	13550	0	90	717	4491.45	-
bnatt400	3600	3600	0	0	5614	1.00	-
bnatt500	4500	4500	0	0	7029	Infeasible	F
bppc4-08	1456	1454	0	2	111	53.00	-
brazil3	23968	23874	94	0	14646	24.00	-
buildingenergy	154978	0	26287	128691	277594	33283.85	-
cbs-cta	24793	2467	0	22326	10112	0.00	-
chromaticindex1024-7	73728	73728	0	0	67583	4.00	-
chromaticindex512-7	36864	36864	0	0	33791	4.00	-
cmflsp50-24-8-8	16392	1392	0	15000	3520	55789389.89	-
CMS750_4	11697	7196	0	4501	16381	252.00	-
co-100	48417	48417	0	0	2187	2639942.06	-
cod105	1024	1024	0	0	1024	-12.00	-
comp07-2idx	17264	17155	109	0	21235	6.00	-

Continued on next page

Table A.1: Statistics for the MIPLIB 2017 instance set. Columns from left to right are: Instance name, number of variables (#Var), binary variables (#Bin), general integers (#Int), continuous variables (#Cont), constraints (#Cons), objective value (Obj), and a marker (F) indicating whether the instance is a feasibility or infeasible instance.

Instance	#Var	#Bin	#Int	#Cont	#Cons	Obj	F
comp21-2idx	10863	10792	71	0	14038	74.00	-
cost266-UUE	4161	171	0	3990	1446	25148940.56	-
cryptanalysiskb128n5obj14	48950	47830	1120	0	98021	Infeasible	F
cryptanalysiskb128n5obj16	48950	47830	1120	0	98021	0.00	F
csched007	1758	1457	0	301	351	351.00	-
csched008	1536	1284	0	252	351	173.00	-
cvs16r128-89	3472	3472	0	0	4633	-97.00	-
dano3_3	13873	69	0	13804	3202	576.34	-
dano3_5	13873	115	0	13758	3202	576.92	-
decomp2	14387	14387	0	0	10765	-160.00	-
drayage-100-23	11090	11025	0	65	4630	103333.87	-
drayage-25-23	11090	11025	0	65	4630	101282.65	-
dws008-01	11096	6608	0	4488	6064	37412.60	-
eil33-2	4516	4516	0	0	32	934.01	-
eilA101-2	65832	65832	0	0	100	880.92	-
enlight_hard	200	100	100	0	100	37.00	-
ex10	17680	0	17680	0	69608	100.00	-
ex9	10404	0	10404	0	40962	81.00	-
exp-1-500-5-5	990	250	0	740	550	65887.00	-
fast0507	63009	63009	0	0	507	174.00	-
fastxgemm-n2r6s0t2	784	48	0	736	5998	230.00	-
fhnw-binpack4-4	520	481	0	39	620	Infeasible	F
fhnw-binpack4-48	3710	3605	0	105	4480	0.00	F
fiball	34219	33960	258	1	3707	138.00	-
gen-ip002	41	0	41	0	24	-4783.73	-
gen-ip054	30	0	30	0	27	6840.97	-
germanrr	10813	5323	5251	239	10779	47095869.65	-
gfd-schedulen180f7d50m30k18	227535	192408	2025	33102	457985	1.00	F
glass-sc	214	214	0	0	6119	23.00	-
glass4	322	302	0	20	396	1200012599.97	-
gmu-35-40	1205	1200	0	5	424	-2406733.37	-
gmu-35-50	1919	1914	0	5	435	-2607958.33	-
graph20-20-1rand	2183	2183	0	0	5587	-9.00	-
graphdraw-domain	254	180	20	54	865	19686.00	-
h80x6320d	12640	6320	0	6320	6558	6382.10	-
highschool1-aigio	320404	319686	718	0	92568	0.00	-
hypothyroid-k1	2602	2601	1	0	5195	-2851.00	-
ic97_potential	728	450	73	205	1046	3942.00	-
icir97_tension	2494	262	573	1659	1203	6375.00	-
irish-electricity	61728	9888	0	51840	104259	3723497.59	-
irp	20315	20315	0	0	39	12159.49	-
istanbul-no-cutoff	5282	30	0	5252	20346	204.08	-
k1mushroom	8211	8210	1	0	16419	-3288.00	-
lectsched-5-obj	21805	21389	416	0	38884	24.00	-
leo1	6731	6730	0	1	593	404227536.16	-
leo2	11100	11099	0	1	593	404077441.12	-
lotsize	2985	1195	0	1790	1920	1480195.00	-
mad	220	200	0	20	51	0.03	-

Continued on next page

Table A.1: Statistics for the MIPLIB 2017 instance set. Columns from left to right are: Instance name, number of variables (#Var), binary variables (#Bin), general integers (#Int), continuous variables (#Cont), constraints (#Cons), objective value (Obj), and a marker (F) indicating whether the instance is a feasibility or infeasible instance.

Instance	#Var	#Bin	#Int	#Cont	#Cons	Obj	F
map10	164547	146	0	164401	328818	-495.00	-
map16715-04	164547	146	0	164401	328818	-111.00	-
markshare2	74	60	0	14	7	1.00	-
markshare_4_0	34	30	0	4	4	1.00	-
mas74	151	150	0	1	13	11801.19	-
mas76	151	150	0	1	12	40005.05	-
mc11	3040	1520	0	1520	1920	11689.00	-
mcsched	1747	1745	0	2	2107	211913.00	-
mik-250-20-75-4	270	75	175	20	195	-52301.00	-
milo-v12-6-r2-40-1	2688	840	0	1848	5628	326481.14	-
momentum1	5174	2349	0	2825	42680	109143.49	-
mushroom-best	8468	8237	118	113	8580	0.06	-
mzzv11	10240	9989	251	0	9499	-21718.00	-
mzzv42z	11717	11482	235	0	10460	-20540.00	-
n2seq36q	22480	22480	0	0	2565	52200.00	-
n3div36	22120	22120	0	0	4484	130800.00	-
n5-3	2550	0	150	2400	1062	8105.00	-
neos-1122047	5100	100	0	5000	57791	161.00	-
neos-1171448	4914	2457	0	2457	13206	-309.00	-
neos-1171737	2340	1170	0	1170	4179	-195.00	-
neos-1354092	13702	13282	420	0	3135	46.00	-
neos-1445765	20617	2150	0	18467	2147	-17783.00	-
neos-1456979	4605	4245	180	180	6770	176.00	-
neos-1582420	10100	10000	100	0	10180	91.00	-
neos-2075418-temuka	122304	122304	0	0	349602	Infeasible	F
neos-2657525-crna	524	146	378	0	342	1.81	-
neos-2746589-doon	50936	50704	224	8	31530	2008.20	-
neos-2978193-inde	20800	64	0	20736	396	-2.39	-
neos-2987310-joes	27837	3051	0	24786	29015	-607702988.30	-
neos-3004026-krka	17030	16900	130	0	12545	0.00	F
neos-3024952-loue	3255	0	3255	0	3705	26756.00	-
neos-3046615-murg	274	240	16	18	498	1600.00	-
neos-3083819-nubu	8644	0	8644	0	4725	6307996.00	-
neos-3216931-puriri	3555	3268	0	287	5989	71320.00	-
neos-3381206-awhea	2375	475	1900	0	479	453.00	-
neos-3402294-bobin	2904	2616	0	288	591076	0.07	-
neos-3402454-bohle	2904	2616	0	288	2897380	Infeasible	F
neos-3555904-turama	37461	37461	0	0	146493	-34.70	-
neos-3627168-kasai	1462	535	0	927	1655	988585.62	-
neos-3656078-kumeu	14870	9755	4455	660	17656	-13172.20	-
neos-3754480-nidda	253	50	0	203	402	12941.74	-
neos-3988577-wolgan	25870	25870	0	0	44662	Infeasible	F
neos-4300652-rahue	33003	20900	0	12103	76992	2.14	-
neos-4338804-snowy	1344	1260	42	42	1701	1471.00	-
neos-4387871-tavua	4004	2000	0	2004	4554	33.38	-
neos-4413714-turia	190402	190201	0	201	2303	45.37	-
neos-4532248-waihi	86842	86841	0	1	167322	61.60	-
neos-4647030-tutaki	12600	7000	0	5600	8382	27265.71	-

Continued on next page

Table A.1: Statistics for the MIPLIB 2017 instance set. Columns from left to right are: Instance name, number of variables (#Var), binary variables (#Bin), general integers (#Int), continuous variables (#Cont), constraints (#Cons), objective value (Obj), and a marker (F) indicating whether the instance is a feasibility or infeasible instance.

Instance	#Var	#Bin	#Int	#Cont	#Cons	Obj	F
neos-4722843-widden	77723	73349	20	4354	113555	25009.66	-
neos-4738912-atrato	6216	1120	5096	0	1947	283627956.60	-
neos-4763324-toguru	53593	53592	0	1	106954	1613.04	-
neos-4954672-berkel	1533	630	0	903	1848	2612710.00	-
neos-5049753-cuanza	242736	8304	0	234432	322248	562.00	-
neos-5052403-cygnet	32868	32868	0	0	38268	182.00	-
neos-5093327-huahum	40640	64	0	40576	51840	6260.00	-
neos-5104907-jarama	345856	9520	0	336336	489818	935.00	-
neos-5107597-kakapo	3114	2976	0	138	6498	3645.00	-
neos-5114902-kasavu	710164	14560	0	695604	961170	655.00	-
neos-5188808-nattai	14544	288	0	14256	29452	0.11	-
neos-5195221-niemur	14546	9792	0	4754	42256	0.00	-
neos-631710	167056	167056	0	0	169576	203.00	-
neos-662469	18235	17907	328	0	1085	184380.00	-
neos-787933	236376	236376	0	0	1897	30.00	-
neos-827175	32504	21350	0	11154	14187	112.00	-
neos-848589	550539	747	0	549792	1484	2351.40	-
neos-860300	1385	1384	0	1	850	3201.00	-
neos-873061	175288	87644	0	87644	93360	113.66	-
neos-911970	888	840	0	48	107	54.76	-
neos-933966	31762	27982	0	3780	12047	318.00	-
neos-950242	5760	5520	240	0	34224	4.00	-
neos-957323	57756	57756	0	0	3757	-237.76	-
neos-960392	59376	59376	0	0	4744	-238.00	-
neos17	535	300	0	235	486	0.15	-
neos5	63	53	0	10	63	15.00	-
neos8	23228	23224	4	0	46324	-3719.00	-
neos859080	160	80	80	0	164	Infeasible	F
net12	14115	1603	0	12512	14021	214.00	-
netdiversion	129180	129180	0	0	119589	242.00	-
nexp-150-20-8-5	20115	17880	0	2235	4620	231.00	-
ns1116954	12648	7482	0	5166	131991	0.00	F
ns1208400	2883	2880	0	3	4289	2.00	-
ns1644855	30200	10000	0	20200	40698	-1524.33	-
ns1760995	17956	17822	0	134	615388	-549.21	-
ns1830653	1629	1458	0	171	2932	20622.00	-
ns1952667	13264	0	13264	0	41	0.00	F
nu25-pr12	5868	5832	36	0	2313	53905.00	-
nursesched-medium-hint03	34248	34170	78	0	14062	115.00	-
nursesched-sprint02	10250	10230	20	0	3522	58.00	-
nw04	87482	87482	0	0	36	16862.00	-
opm2-z10-s4	6250	6250	0	0	160633	-33269.00	-
p200x1188c	2376	1188	0	1188	1388	15078.00	-
peg-solitaire-a3	4552	4552	0	0	4587	1.00	-
pg	2700	100	0	2600	125	-8674.34	-
pg5_34	2600	100	0	2500	225	-14339.35	-
physiciansched3-3	79555	72141	0	7414	266227	2623271.33	-
physiciansched6-2	111827	109346	0	2481	168336	49324.00	-

Continued on next page

Table A.1: Statistics for the MIPLIB 2017 instance set. Columns from left to right are: Instance name, number of variables (#Var), binary variables (#Bin), general integers (#Int), continuous variables (#Cont), constraints (#Cons), objective value (Obj), and a marker (F) indicating whether the instance is a feasibility or infeasible instance.

Instance	#Var	#Bin	#Int	#Cont	#Cons	Obj	F
piperout-08	10399	10245	130	24	14589	125055.00	-
piperout-27	11659	11514	121	24	18442	8124.00	-
pk1	86	55	0	31	45	11.00	-
proteindesign121hz512p9	159145	159054	91	0	301	1473.00	-
proteindesign122trx11p8	127326	127248	78	0	254	1747.00	-
qap10	4150	4150	0	0	1820	340.00	-
radiationm18-12-05	40623	14688	11247	14688	40935	17566.00	-
radiationm40-10-02	172013	62400	47213	62400	173603	155328.00	-
rail01	117527	117527	0	0	46843	-70.57	-
rail02	270869	270869	0	0	95791	-200.45	-
rail507	63019	63009	0	10	509	174.00	-
ran14x18-disj-8	504	252	0	252	447	3712.00	-
rd-rplusc-21	622	457	0	165	125899	165395.28	-
reblock115	1150	1150	0	0	4735	-36800603.23	-
rmatr100-p10	7359	100	0	7259	7260	423.00	-
rmatr200-p5	37816	200	0	37616	37617	4521.00	-
rocI-4-11	6839	5192	1016	631	10883	-6020203.00	-
rocII-5-11	11523	11341	0	182	26897	-6.68	-
rococoB10-011000	4456	4320	136	0	1667	19449.00	-
rococoC10-001000	3117	2993	124	0	1293	11460.00	-
roi2alpha3n4	6816	6642	0	174	1251	-63.21	-
roi5alpha10n8	106150	105950	0	200	4665	-52.32	-
roll3000	1166	246	492	428	2295	12890.00	-
s100	364417	364417	0	0	14733	-0.17	-
s250r10	273142	273139	0	3	10962	-0.17	-
satellites2-40	35378	34324	0	1054	20916	-19.00	-
satellites2-60-fs	35378	34324	0	1054	16516	-19.00	-
savsched1	328575	252731	0	75844	295989	3217.70	-
sct2	5885	2872	0	3013	2151	-230.99	-
seymour	1372	1372	0	0	4944	423.00	-
seymour1	1372	451	0	921	4944	410.76	-
sing326	55156	40010	0	15146	50781	7753674.85	-
sing44	59708	43524	0	16184	54745	8128831.18	-
snp-02-004-104	228350	167	167	228016	126512	586803238.66	-
sorrell3	1024	1024	0	0	169162	-16.00	-
sp150x300d	600	300	0	300	450	69.00	-
sp97ar	14101	14101	0	0	1761	660705645.76	-
sp98ar	15085	15085	0	0	1435	529740623.20	-
splice1k1	3253	3252	1	0	6505	-394.00	-
square41	62234	62197	37	0	40160	15.00	-
square47	95030	94987	43	0	61591	16.00	-
supportcase10	14770	14770	0	0	165684	7.00	-
supportcase12	799616	0	200	799416	166781	-7559.53	-
supportcase18	13410	13410	0	0	240	48.00	-
supportcase19	1429098	1311292	117806	0	10713	12677206.00	-
supportcase22	7129	7129	0	0	260602	110.0*	-
supportcase26	436	396	0	40	870	1745.12	-
supportcase33	20203	20102	101	0	20489	-345.00	-

Continued on next page

Table A.1: Statistics for the MIPLIB 2017 instance set. Columns from left to right are: Instance name, number of variables (#Var), binary variables (#Bin), general integers (#Int), continuous variables (#Cont), constraints (#Cons), objective value (Obj), and a marker (F) indicating whether the instance is a feasibility or infeasible instance.

Instance	#Var	#Bin	#Int	#Cont	#Cons	Obj	F
supportcase40	16440	2000	0	14440	38192	24256.31	-
supportcase42	19466	0	1026	18440	18439	7.76	-
supportcase6	130052	130051	1	0	771	51906.48	-
supportcase7	138844	451	14	138379	6532	-1132.22	-
swath1	6805	2306	0	4499	884	379.07	-
swath3	6805	2706	0	4099	884	397.76	-
tbfp-network	72747	72747	0	0	2436	24.16	-
thor50dday	106261	53131	0	53130	53360	40417.00	-
timtab1	397	77	94	226	171	764772.00	-
tr12-30	1080	360	0	720	750	130596.00	-
traininstance2	12890	5278	2602	5010	15603	71820.00	-
traininstance6	10218	4154	2056	4008	12309	28290.00	-
trento1	7687	6415	0	1272	1265	5189487.00	-
triptim1	30055	20456	9592	7	15706	22.87	-
uccase12	62529	9072	0	53457	121161	11507.41	-
uccase9	33242	8064	0	25178	49565	10993.13	-
uct-subprob	2256	379	0	1877	1973	314.00	-
unitcal_7	25755	2856	0	22899	48939	19635558.24	-
var-smallemery-m6j6	5608	5606	0	2	13416	-149.38	-
wachplan	3361	3360	1	0	1553	-8.00	-

Chapter B

Root Gap Closed Experiment Results

Table B.1: Comparison of root node gap closed after adding a single round of Gomory cuts versus a single round of V-polyhedral cuts (VPCs). Also reported are the number of fractional variables, the number of Gomory cuts and VPCs added, and the gap closed implied by the disjunctive lower bound, along with (when applicable) the reasons for VPC generation failure. Possible failure reasons are: M (memory limit), Z (encountered non-basic variable fixed to zero), F (failure to prove PRLP or tightened PRLP feasibility), B (hit time limit while generating the partial branch-and-bound tree), C (hit time limit while collecting points and rays), L (LP error), and N (not run).

		# cuts			Gap Closed (%)		
Instance	# Frac. Vars.	Gomory	VPC	Failure	Gomory	VPC	Disjunctive
30n20b8	124	118	0	F	10.69	0.00	2.00
50v-10	29	29	29	-	41.74	4.94	12.20
CMS750_4	1848	1848	8	-	0.00	0.00	0.00
academictimetablesmall	912	912	0	F	0.00	0.00	100.00
air05	222	218	128	-	4.56	40.45	42.59
app1-1	669	652	4	-	4.88	0.00	79.67
app1-2	7625	2000	0	B	0.09	0.00	0.00
assign1-5-8	114	134	114	-	22.48	12.00	12.45
atlanta-ip	1665	247	554	-	0.66	0.00	0.00
b1c1s1	247	239	152	-	24.79	0.21	4.70
bab2	687	687	0	C	42.70	0.00	0.00
bab6	1130	1130	0	C	42.94	0.00	0.00
beasleyC3	148	148	148	-	1.08	1.79	2.97
binkar10_1	38	38	38	-	2.51	8.34	9.32
blp-ar98	129	128	69	-	35.55	0.00	5.55
blp-ic98	54	52	54	-	34.86	1.00	10.29
bnatt400	546	541	0	F	0.00	0.00	0.00
bppc4-08	49	49	0	F	29.08	0.00	0.00
brazil3	917	898	0	Z	6.67	0.00	0.00
buildingenergy	7943	2000	0	C	0.03	0.00	0.00
cbs-cta	170	154	0	F	0.00	0.00	100.00
chromaticindex1024-7	49120	2000	0	B	0.00	0.00	0.00

Continued on next page

		No. cuts			Gap Closed (%)		
Instance	# Frac. Vars.	Gomory	VPC	Failure	Gomory	VPC	Disjunctive
chromaticindex512-7	24213	2000	0	F	0.00	0.00	0.00
cmflsp50-24-8-8	491	120	0	F	2.91	0.00	16.12
co-100	215	102	0	L	0.50	0.00	0.00
cod105	245	232	4	-	0.81	7.05	100.00
comp07-2idx	876	872	0	F	0.00	0.00	0.00
comp21-2idx	699	698	0	F	3.60	0.00	0.00
cost266-UUE	56	55	56	-	23.62	11.57	16.40
csched007	84	110	84	-	10.39	7.23	8.95
csched008	83	142	0	F	0.00	0.00	0.00
cvs16r128-89	3210	2000	0	F	4.84	0.00	3.46
dano3_3	12	0	12	-	0.00	4.55	100.00
dano3_5	25	1	0	F	1.45	0.00	49.86
decomp2	228	228	228	-	0.00	100.00	100.00
drayage-100-23	64	64	0	F	0.00	0.00	0.00
drayage-25-23	125	125	0	F	0.00	0.00	0.00
dws008-01	28	28	28	-	1.83	0.62	2.40
eil33-2	30	29	30	-	4.43	5.98	16.10
eilA101-2	71	64	0	F	2.00	0.00	11.09
enlight_hard	43	43	0	Z	4.65	0.00	0.00
ex10	0	0	0	N	0.00	0.00	0.00
ex9	0	0	0	N	0.00	0.00	0.00
exp-1-500-5-5	133	130	70	-	21.98	1.79	10.59
fast0507	259	257	2	-	1.92	8.74	10.68
fastxgemm-n2r6s0t2	22	110	0	F	0.00	0.00	0.00
fiball	272	273	272	-	100.00	0.00	0.00
gen-ip002	18	18	18	-	2.15	8.55	8.84
gen-ip054	15	15	15	-	1.71	6.85	7.02
germanrr	209	209	0	L	10.18	0.00	0.00
glass-sc	101	98	101	-	0.65	28.96	29.09
glass4	71	71	68	-	0.00	0.00	0.00
gmu-35-40	11	7	11	-	0.07	0.00	0.00
gmu-35-50	16	16	16	-	0.00	0.00	0.00
graph20-20-1rand	48	48	48	-	0.00	1.74	1.75
graphdraw-domain	93	90	93	-	0.00	0.08	0.84
h80x6320d	116	108	72	-	31.11	3.61	13.25
hypothyroid-k1	143	120	4	-	2.73	14.24	100.00
ic97_potential	311	311	311	-	0.48	0.00	0.00
icir97_tension	578	578	69	-	19.76	0.00	0.00
irish-electricity	4456	2000	0	B	18.79	0.00	0.00
irp	17	17	17	-	19.77	12.73	28.69
istanbul-no-cutoff	13	13	13	-	4.00	12.01	22.33
k1mushroom	296	249	57	-	0.16	0.58	21.33
lectsched-5-obj	2323	2000	71	-	8.20	0.00	0.00
leo1	65	65	0	F	10.48	0.00	4.86
lotsize	524	473	69	-	11.59	0.20	1.83
mad	18	18	0	F	0.00	0.00	0.00
map10	65	64	0	F	1.22	0.00	39.13
map16715-04	69	66	0	F	0.22	0.00	24.00
markshare2	7	7	0	F	0.00	0.00	0.00
markshare_4_0	4	4	0	F	0.00	0.00	0.00
mas74	12	12	12	-	6.67	10.81	11.50

Continued on next page

Instance	# Frac. Vars.	No. cuts Gomory	VPC	Failure	Gap Closed (%) Gomory	VPC	Disjunctive
mas76	11	11	11	-	6.42	11.98	12.39
mc11	363	206	363	-	0.44	0.26	0.53
mcsched	1259	1259	810	-	0.04	2.70	4.12
mik-250-20-75-4	75	75	75	-	53.83	13.37	19.40
milo-v12-6-r2-40-1	358	0	358	-	0.00	0.02	1.10
momentum1	231	113	0	F	56.93	0.00	0.26
mushroom-best	26	0	5	-	0.00	0.01	3.22
mzzv11	757	762	0	Z	23.17	0.00	0.00
mzzv42z	722	721	9	-	18.04	1.01	2.31
n2seq36q	277	277	0	F	0.00	0.00	0.00
n3div36	24	23	24	-	10.15	5.60	6.59
n5-3	34	34	34	-	17.21	3.55	27.42
neos-1171448	109	109	0	F	0.00	0.00	100.00
neos-1171737	73	73	0	F	0.00	0.00	100.00
neos-1354092	907	874	0	F	47.88	0.00	0.00
neos-1445765	145	142	145	-	1.32	36.42	36.96
neos-1456979	119	119	73	-	0.00	0.00	4.42
neos-1582420	292	292	292	-	3.95	26.74	28.17
neos-2657525-crna	64	59	0	F	0.00	0.00	0.00
neos-2746589-doon	254	249	0	Z	9.48	0.00	0.00
neos-2978193-inde	32	32	0	Z	0.00	0.00	0.00
neos-2987310-joes	0	0	0	N	0.00	0.00	0.00
neos-3024952-loue	458	458	458	-	12.54	0.00	0.00
neos-3046615-murg	86	14	69	-	0.00	0.00	1.73
neos-3083819-nubu	33	33	0	L	59.11	0.00	0.00
neos-3216931-puriri	647	463	0	F	0.64	0.00	1.42
neos-3381206-awhea	403	403	132	-	0.00	0.00	0.00
neos-3402294-bobin	171	162	0	F	0.00	0.00	0.00
neos-3555904-turama	1396	491	0	M	0.00	0.00	0.00
neos-3627168-kasai	146	124	146	-	19.90	0.68	0.84
neos-3656078-kumeu	2437	801	3	-	7.67	0.00	0.00
neos-3754480-nidda	41	0	41	-	0.00	21.32	22.29
neos-4300652-rahue	104	104	17	-	0.00	0.00	0.00
neos-4338804-snowy	353	95	102	-	0.00	0.00	0.00
neos-4387871-tavua	34	34	34	-	1.04	8.02	12.66
neos-4413714-turia	2016	1948	0	L	20.45	0.00	0.00
neos-4532248-waihi	4972	268	0	C	0.23	0.00	0.00
neos-4647030-tutaki	799	799	69	-	97.20	0.00	0.00
neos-4722843-widden	25389	2000	0	C	3.71	0.00	3.95
neos-4738912-atrato	233	224	68	-	12.69	0.00	35.99
neos-4763324-toguru	1563	1512	0	C	0.80	0.00	0.00
neos-4954672-berkel	58	58	58	-	10.74	0.15	2.83
neos-5049753-cuanza	230	230	0	C	18.47	0.00	0.00
neos-5052403-cygnet	695	692	2	-	29.56	7.78	15.63
neos-5093327-huahum	64	60	64	-	0.40	0.17	25.19
neos-5104907-jarama	952	304	0	C	41.22	0.00	0.00
neos-5107597-kakapo	1519	1273	71	-	0.00	0.00	0.00
neos-5114902-kasavu	259	259	0	B	23.60	0.00	0.00
neos-5188808-nattai	23	23	23	-	0.00	0.00	0.00
neos-5195221-niemur	1138	1135	40	-	0.00	0.00	0.00
neos-631710	1139	1137	0	C	0.00	0.00	0.00

Continued on next page

		No. cuts			Gap Closed (%)		
Instance	# Frac. Vars.	Gomory	VPC	Failure	Gomory	VPC	Disjunctive
neos-662469	348	311	2	-	21.56	6.60	9.26
neos-787933	22	22	10	-	0.00	100.00	100.00
neos-827175	0	0	0	N	0.00	0.00	0.00
neos-848589	617	12	0	M	0.00	0.00	0.00
neos-860300	106	70	106	-	1.05	11.83	17.59
neos-873061	92	82	0	C	35.19	0.00	0.00
neos-911970	55	55	55	-	0.00	0.00	0.00
neos-933966	1275	1275	0	F	0.00	0.00	99.99
neos-950242	116	115	5	-	0.00	0.00	0.00
neos-957323	123	123	0	F	16.36	0.00	0.00
neos-960392	404	387	2	-	0.00	100.00	100.00
neos17	171	7	171	-	0.01	0.29	1.19
neos5	35	35	35	-	11.57	23.21	23.21
neos8	193	193	167	-	100.00	0.00	100.00
net12	374	374	42	-	2.94	0.00	0.26
netdiversion	3902	2000	0	C	6.89	0.00	0.00
nexp-150-20-8-5	74	70	74	-	4.48	0.29	0.71
ns1208400	324	323	0	F	0.00	0.00	0.00
ns1644855	343	337	0	B	0.00	0.00	0.00
ns1760995	1060	189	0	B	0.39	0.00	0.00
ns1830653	167	170	167	-	12.78	14.41	14.41
nu25-pr12	37	37	37	-	57.39	2.11	13.98
nursesched-medium-hint03	1224	1205	0	F	4.15	0.00	0.00
nursesched-sprint02	448	448	242	-	44.19	0.00	0.00
nw04	6	6	6	-	30.71	4.17	33.49
opm2-z10-s4	5584	3	0	B	0.01	0.00	0.00
p200x1188c	5	3	5	-	0.43	8.59	9.49
peg-solitaire-a3	1005	979	0	Z	0.00	0.00	0.00
pg	93	91	6	-	24.29	0.25	8.97
pg5_34	88	88	5	-	23.45	0.26	6.94
physiciansched3-3	1658	1565	0	F	14.05	0.00	0.00
physiciansched6-2	974	972	0	F	0.02	0.00	0.00
piperout-08	1595	1583	0	Z	0.00	0.00	0.00
piperout-27	2480	2000	70	-	31.35	0.00	33.00
pk1	15	15	0	F	0.00	0.00	0.00
proteindesign121hz512p9	209	214	101	-	10.40	0.00	1.89
proteindesign122trx11p8	163	163	0	Z	20.41	0.00	0.00
qap10	1218	1048	28	-	11.27	37.56	100.00
radiationm18-12-05	945	949	71	-	37.66	3.59	3.59
radiationm40-10-02	2964	2000	0	F	21.59	0.00	0.00
rail01	7535	2000	0	B	16.23	0.00	0.00
rail02	10824	2000	0	B	19.90	0.00	0.00
rail507	254	252	2	-	1.98	8.60	11.07
ran14x18-disj-8	86	80	86	-	0.15	7.40	7.49
reblock115	878	92	0	F	1.33	0.00	6.30
rmatr100-p10	51	51	51	-	1.68	39.27	69.31
rmatr200-p5	66	60	0	F	0.10	0.00	39.45
rocI-4-11	421	420	52	-	0.00	0.00	0.00
rocII-5-11	181	181	181	-	0.07	0.04	0.40
rococoB10-011000	261	256	261	-	9.85	0.74	0.88
rococoC10-001000	188	158	188	-	25.04	3.42	5.33

Continued on next page

		No. cuts			Gap Closed (%)		
Instance	# Frac. Vars.	Gomory	VPC	Failure	Gomory	VPC	Disjunctive
roi2alpha3n4	37	30	7	-	15.49	1.65	16.05
roi5alpha10n8	89	67	6	-	11.41	2.87	9.78
roll3000	200	202	79	-	4.06	0.03	0.54
s100	198	192	0	B	3.47	0.00	0.00
s250r10	294	294	0	C	20.24	0.00	0.00
satellites2-40	1767	1714	0	F	0.00	0.00	0.00
satellites2-60-fs	1704	1600	0	B	0.00	0.00	0.00
savsched1	15794	2000	0	B	0.04	0.00	0.00
sct2	49	45	0	F	41.78	0.00	0.00
seymour	544	542	544	-	5.28	11.60	11.62
seymour1	135	133	135	-	8.39	22.56	22.87
sing326	934	814	110	-	12.05	0.45	2.65
sing44	760	723	108	-	6.34	0.88	4.06
snp-02-004-104	68	62	0	C	23.31	0.00	0.00
sorrell3	1024	1024	653	-	0.50	6.35	6.35
sp150x300d	42	30	42	-	1.47	1.39	4.34
sp97ar	194	191	0	F	12.05	0.00	7.21
sp98ar	157	154	0	L	16.17	0.00	0.00
splice1k1	290	207	24	-	0.01	0.03	0.34
square41	342	264	0	B	3.47	0.00	0.00
square47	347	281	0	B	4.23	0.00	0.00
supportcase10	7705	2000	0	B	2.00	0.00	0.00
supportcase12	190	188	7	-	0.00	0.00	0.05
supportcase18	100	99	0	F	0.00	0.00	0.00
supportcase22	62	62	0	B	0.00	0.00	0.00
supportcase26	236	31	74	-	0.00	0.00	0.00
supportcase33	269	269	11	-	22.03	21.62	35.14
supportcase40	38	38	0	F	3.65	0.00	26.83
supportcase42	151	4	0	Z	0.00	0.00	0.00
supportcase6	140	140	0	Z	99.78	0.00	0.00
supportcase7	253	253	10	-	19.23	0.09	18.13
swath1	12	12	12	-	9.89	0.05	31.64
swath3	16	16	16	-	0.75	4.91	14.59
tbfp-network	208	207	0	M	27.28	0.00	0.00
thor50dday	54	54	0	C	0.01	0.00	0.00
timtab1	109	109	85	-	12.74	2.05	6.79
tr12-30	326	326	326	-	58.64	1.59	1.99
traininstance2	110	34	0	F	0.00	0.00	0.00
traininstance6	11	8	11	-	0.00	0.00	0.00
trento1	482	438	0	F	4.14	0.00	9.83
triptim1	3454	2000	0	F	100.00	0.00	0.00
uccase12	81	80	0	C	21.18	0.00	0.00
uccase9	422	225	76	-	14.41	0.13	0.22
uct-subprob	210	821	210	-	1.16	4.33	8.09
unitcal_7	719	700	71	-	8.10	0.35	10.46
var-smallemery-m6j6	396	395	396	-	3.37	8.88	16.53
wachplan	285	285	0	F	0.00	0.00	0.00

Table B.2: Comparison of root node gap closed over instance set `Set B` after SCIP default cuts, applying VPC before SCIP default cuts (VPC+SCIP), and applying VPC after SCIP default cuts (SCIP+VPC), along with the reasons for failures in SCIP+VPC (when applicable). Possible failure reasons are: M (memory limit), Z (encountered non-basic variable fixed to zero), F (failure to prove PRLP or tightened PRLP feasibility), B (hit time limit while generating the partial branch-and-bound tree), C (hit time limit while collecting points and rays), L (LP error), and N (not run). Only one instance failed in the VPC+SCIP configuration: `rmatr100_p10` (F).

	Gap Closed (%)			# Cuts		
Instance	Default	VPC+SCIP	SCIP+VPC	VPC+SCIP	SCIP+VPC	Failure
50v-10	79.93	81.55	81.45	200	102	-
CMS750_4	0.00	0.00	0.00	8	0	F
air05	23.98	41.52	23.98	121	142	-
app1-1	5.29	5.93	5.29	4	6	-
assign1-5-8	26.11	34.07	28.66	200	200	-
atlanta-ip	1.76	1.89	1.76	200	0	F
b1c1s1	75.52	75.73	76.00	158	0	F
beasleyC3	99.41	99.40	99.41	200	70	-
binkar10_1	71.87	73.16	80.23	200	200	-
blp-ar98	61.04	61.89	62.05	69	109	-
blp-ic98	21.86	29.52	28.70	200	127	-
cod105	45.25	51.88	51.78	4	16	-
cost266-UUE	20.67	11.64	20.67	200	0	L
csched007	42.45	42.45	45.15	200	200	-
dano3_3	9.86	12.87	17.18	13	12	-
decomp2	0.00	0.00	0.00	200	0	F
dws008-01	17.98	19.31	22.85	200	144	-
eil33-2	5.28	8.89	9.23	200	200	-
exp-1-500-5-5	100.00	100.00	100.00	70	0	F
fast0507	4.09	9.83	4.09	2	2	-
fiball	0.00	0.00	0.00	200	200	-
gen-ip002	5.57	8.51	8.62	200	200	-
gen-ip054	4.24	6.90	7.83	200	200	-
glass-sc	1.34	32.73	32.73	200	200	-
glass4	0.00	0.00	0.00	68	0	L
gmu-35-40	9.35	9.35	9.35	200	200	-
gmu-35-50	0.00	0.01	0.00	200	200	-
graph20-20-1rand	0.00	33.33	0.00	200	0	Z
graphdraw-domain	4.49	4.14	4.49	200	0	L
h80x6320d	96.52	97.44	96.52	72	100	-
hypothyroid-k1	72.38	100.00	75.62	4	8	-
ic97_potential	4.42	0.00	1.35	200	0	F
icir97_tension	76.75	76.65	70.47	69	0	F
irp	49.06	60.88	57.99	200	200	-
istanbul-no-cutoff	42.66	42.72	34.37	32	0	F
k1mushroom	0.56	1.34	0.56	66	0	F
lectsched-5-obj	11.48	11.48	11.48	71	68	-
lotsize	98.37	98.19	98.37	69	0	F
mas74	8.56	11.18	12.73	200	200	-
mas76	8.73	11.89	14.30	200	200	-
mc11	98.52	98.55	98.52	200	78	-
mcsched	0.23	0.23	1.27	200	200	-
mik-250-20-75-4	81.19	80.65	83.05	200	200	-

Continued on next page

Instance	Gap Closed (%) Default	VPC+SCIP	SCIP+VPC	# Cuts VPC+SCIP	SCIP+VPC	Failure
milo-v12-6-r2-40-1	29.86	31.11	29.86	200	0	F
mushroom-best	0.00	0.00	0.00	4	5	-
mzzv42z	75.76	78.32	75.76	9	0	F
n3div36	55.23	55.22	60.45	197	120	-
n5-3	78.78	82.79	81.85	200	123	-
neos-1445765	16.11	36.86	52.92	200	200	-
neos-1456979	45.45	39.77	45.45	73	200	-
neos-1582420	66.35	60.10	66.35	200	196	-
neos-3024952-loue	38.79	35.59	38.83	200	73	-
neos-3046615-murg	8.89	9.05	9.02	69	200	-
neos-3381206-awhea	0.00	0.03	97.45	132	69	-
neos-3627168-kasai	29.38	28.11	33.48	200	0	F
neos-3656078-kumeu	13.14	13.55	12.83	3	3	-
neos-3754480-nidda	0.00	0.00	0.00	200	200	-
neos-4300652-rahue	1.73	1.32	1.84	17	5	-
neos-4338804-snowy	0.00	0.00	0.00	102	0	F
neos-4387871-tavua	64.08	71.29	64.08	200	0	F
neos-4647030-tutaki	99.86	99.86	99.86	69	0	F
neos-4738912-atrato	47.78	55.51	47.78	68	0	L
neos-4954672-berkel	65.53	65.13	66.59	91	200	-
neos-5052403-cygnet	1.10	13.90	1.10	2	0	F
neos-5093327-huahum	42.57	42.75	42.57	70	0	F
neos-5107597-kakapo	1.73	1.73	0.01	71	0	F
neos-5188808-nattai	0.00	0.00	0.00	200	0	Z
neos-5195221-niemur	0.00	0.00	0.00	40	11	-
neos-662469	68.76	68.76	72.78	2	2	-
neos-787933	0.00	0.00	0.00	10	0	F
neos-860300	62.53	62.58	89.11	200	200	-
neos-911970	95.29	92.55	95.29	128	200	-
neos-950242	0.00	0.00	0.00	16	9	-
neos-960392	0.00	0.00	0.00	2	0	F
neos17	58.51	57.95	75.45	200	200	-
neos5	19.47	23.21	37.50	200	200	-
neos8	100.00	100.00	100.00	167	0	F
net12	10.34	8.70	9.00	70	0	F
nexp-150-20-8-5	99.48	99.47	99.48	200	0	M
ns1830653	33.53	33.53	33.81	200	200	-
nu25-pr12	84.83	84.82	87.20	200	200	-
nursesched-sprint02	86.47	91.00	86.47	200	0	F
nw04	1.33	7.34	23.26	200	200	-
p200x1188c	100.00	100.00	100.00	200	0	F
pg	99.82	99.82	99.88	6	200	-
pg5_34	98.81	98.82	98.91	5	200	-
piperout-27	57.21	59.18	57.21	70	0	Z
proteindesign121hz512p9	22.72	23.53	22.72	126	0	M
qap10	0.00	37.60	76.92	28	44	-
radiationm18-12-05	35.31	42.96	35.31	71	200	-
rail507	4.08	9.19	4.08	2	2	-
ran14x18-disj-8	8.37	7.57	18.68	200	200	-
rmatr100-p10	4.72	4.72	4.72	0	0	F
rocI-4-11	0.00	0.00	0.00	52	0	F

Continued on next page

	Gap Closed (%)			# Cuts		
Instance	Default	VPC+SCIP	SCIP+VPC	VPC+SCIP	SCIP+VPC	Failure
rocII-5-11	0.40	0.33	0.40	200	0	Z
rococoB10-011000	46.22	46.22	46.22	200	0	F
rococoC10-001000	69.01	69.28	69.01	200	16	-
roi2alpha3n4	36.40	37.80	36.40	7	30	-
roi5alpha10n8	45.93	46.92	45.93	6	0	M
roll3000	80.88	77.98	81.53	79	200	-
seymour	33.69	29.97	38.93	200	200	-
seymour1	35.86	36.98	46.02	200	200	-
sing326	52.65	60.76	52.65	165	0	F
sing44	25.92	23.52	25.92	108	0	F
sorrell3	99.27	99.27	99.27	200	0	B
sp150x300d	82.81	82.81	85.68	200	70	-
splice1k1	0.03	0.03	0.03	24	0	N
supportcase12	0.00	0.00	0.00	7	5	-
supportcase26	30.29	30.23	32.53	74	200	-
supportcase33	41.03	41.03	41.03	10	6	-
supportcase7	96.33	94.15	95.57	10	0	F
swath1	14.01	23.03	26.16	200	200	-
swath3	16.56	16.47	23.16	200	200	-
timtab1	50.89	55.05	50.49	85	200	-
tr12-30	99.72	99.73	99.77	200	200	-
traininstance6	0.00	0.00	0.00	46	0	F
uccase9	27.53	20.77	27.54	76	90	-
uct-subprob	36.96	36.96	37.70	200	200	-
unitcal_7	83.73	89.11	86.98	71	0	F
var-smallemery-m6j6	0.00	0.00	0.00	200	200	-

Table B.3: Comparison of root node gap closed using Barrier without crossover (NX) and Barrier with crossover (WX) for solving PRLP. Each method reports the number of cuts added, the percentage of root node gap closed, along with, when applicable, failure reasons. Possible failure reasons are: M (memory limit), Z (encountered non-basic variable fixed to zero), F (failure to prove PRLP or tightened PRLP feasibility), B (hit time limit while generating the partial branch-and-bound tree), C (hit time limit while collecting points and rays), L (LP error), and N (not run).

		# Cuts		Gap Closed (%)		Failure	
Instance	# Frac. Vars.	NX	WX	NX	WX	NX	WX
30n20b8	124	0	0	0.00	0.00	F	F
50v-10	29	29	29	1.14	2.02	-	-
CMS750_4	1848	46	8	0.00	0.00	-	-
academictimetablesmall	912	0	0	0.00	0.00	F	F
air05	222	114	78	20.02	38.47	-	-
app1-1	669	669	3	0.00	0.00	-	-
app1-2	7625	0	0	0.00	0.00	B	B
assign1-5-8	114	114	114	8.37	12.05	-	-
atlanta-ip	1665	444	357	0.00	0.00	-	-
b1c1s1	247	247	94	0.50	0.74	-	-
bab2	687	0	0	0.00	0.00	C	C
bab6	1130	0	0	0.00	0.00	F	C
beasleyC3	148	148	148	1.13	1.95	-	-

Continued on next page

Instance	# Frac. Vars.	# Cuts NX	# Cuts WX	Gap Closed (%) NX	Gap Closed (%) WX	Failure NX	Failure WX
binkar10_1	38	38	38	3.69	7.76	-	-
blp-ar98	129	129	4	0.96	0.00	-	-
blp-ic98	54	54	54	7.85	7.83	-	-
bnatt400	546	0	0	0.00	0.00	F	F
bppc4-08	49	0	0	0.00	0.00	F	F
brazil3	917	0	0	0.00	0.00	Z	Z
buildingenergy	7943	0	0	0.00	0.00	C	C
cbs-cta	170	0	0	0.00	0.00	F	F
chromaticindex1024-7	49120	0	0	0.00	0.00	B	B
chromaticindex512-7	24213	0	0	0.00	0.00	F	F
cmflsp50-24-8-8	491	0	2	0.00	8.68	L	-
co-100	215	215	215	0.01	0.04	-	-
cod105	245	67	67	0.09	0.09	-	-
comp07-2idx	876	0	0	0.00	0.00	F	F
comp21-2idx	699	0	0	0.00	0.00	F	F
cost266-UUE	56	56	56	4.60	10.48	-	-
csched007	84	84	84	4.79	5.77	-	-
csched008	83	0	0	0.00	0.00	F	F
cvs16r128-89	3210	2	0	0.00	0.00	-	F
dano3_3	12	10	8	3.22	4.19	-	-
dano3_5	25	2	0	0.00	0.00	-	F
decomp2	228	228	228	0.00	0.00	-	-
drayage-100-23	64	0	0	0.00	0.00	F	F
drayage-25-23	125	0	0	0.00	0.00	F	F
dws008-01	28	28	28	0.57	1.78	-	-
eil33-2	30	30	30	0.00	0.57	-	-
eilA101-2	71	2	2	0.00	2.01	-	-
enlight_hard	43	0	0	0.00	0.00	Z	Z
ex10	0	0	0	0.00	0.00	N	N
ex9	0	0	0	0.00	0.00	N	N
exp-1-500-5-5	133	133	7	0.14	0.26	-	-
fast0507	259	2	2	1.80	2.32	-	-
fastxgemm-n2r6s0t2	22	0	0	0.00	0.00	F	F
fiball	272	272	272	0.00	0.00	-	-
gen-ip002	18	18	18	8.27	8.45	-	-
gen-ip054	15	15	15	6.75	6.81	-	-
germanrr	209	0	135	0.00	1.70	L	-
glass-sc	101	101	101	28.70	28.96	-	-
glass4	71	71	31	0.00	0.00	-	-
gmu-35-40	11	11	11	0.00	0.00	-	-
gmu-35-50	16	16	16	0.00	0.00	-	-
graph20-20-1rand	48	48	48	1.12	1.75	-	-
graphdraw-domain	93	0	93	0.00	0.00	F	-
h80x6320d	116	116	22	2.05	3.96	-	-
hypothyroid-k1	143	3	3	13.67	13.67	-	-
ic97_potential	311	311	311	0.00	0.00	-	-
icir97_tension	578	578	6	0.00	0.00	-	-
irish-electricity	4456	0	0	0.00	0.00	B	C
irp	17	17	17	0.00	0.03	-	-
istanbul-no-cutoff	13	13	13	0.00	7.28	-	-

Continued on next page

Instance	# Frac. Vars.	# Cuts NX	# Cuts WX	Gap Closed (%) NX	Gap Closed (%) WX	Failure NX	Failure WX
k1mushroom	296	0	0	0.00	0.00	F	F
lectsched-5-obj	2323	185	69	0.00	0.00	-	-
leo1	65	0	0	0.00	0.00	F	L
lotsize	524	0	6	0.00	0.30	L	-
mad	18	0	0	0.00	0.00	F	F
map10	65	2	2	0.00	6.78	-	-
map16715-04	69	2	0	0.00	0.00	-	F
markshare2	7	0	0	0.00	0.00	F	F
markshare_4_0	4	0	0	0.00	0.00	F	F
mas74	12	12	12	10.71	10.94	-	-
mas76	11	11	11	11.61	11.93	-	-
mc11	363	363	363	0.15	0.26	-	-
mcsched	1259	547	341	1.24	2.43	-	-
mik-250-20-75-4	75	75	75	5.34	14.97	-	-
milo-v12-6-r2-40-1	358	214	73	0.00	0.01	-	-
momentum1	231	0	0	0.00	0.00	L	F
mushroom-best	26	26	26	0.00	0.01	-	-
mzzv11	757	0	0	0.00	0.00	Z	Z
mzzv42z	722	722	5	0.00	0.15	-	-
n2seq36q	277	0	0	0.00	0.00	F	F
n3div36	24	24	24	3.75	5.28	-	-
n5-3	34	34	34	8.63	17.17	-	-
neos-1171448	109	0	0	0.00	0.00	F	F
neos-1171737	73	0	0	0.00	0.00	F	F
neos-1354092	907	0	0	0.00	0.00	F	F
neos-1445765	145	145	145	34.55	36.54	-	-
neos-1456979	119	119	7	0.00	2.29	-	-
neos-1582420	292	292	261	16.89	27.38	-	-
neos-2657525-crna	64	0	0	0.00	0.00	F	F
neos-2746589-doon	254	0	0	0.00	0.00	Z	Z
neos-2978193-inde	32	0	0	0.00	0.00	Z	Z
neos-2987310-joes	0	0	0	0.00	0.00	N	N
neos-3024952-loue	458	458	458	0.00	0.00	-	-
neos-3046615-murg	86	86	22	0.00	0.79	-	-
neos-3083819-nubu	33	0	33	0.00	7.64	L	-
neos-3216931-puriri	647	0	0	0.00	0.00	F	F
neos-3381206-awhea	403	81	67	0.00	0.00	-	-
neos-3402294-bobin	171	0	0	0.00	0.00	F	F
neos-3627168-kasai	146	146	146	0.00	0.72	-	-
neos-3656078-kumeu	2437	0	0	0.00	0.00	F	F
neos-3754480-nidda	41	41	41	16.27	20.05	-	-
neos-4300652-rahue	104	104	11	0.00	0.00	-	-
neos-4338804-snowy	353	211	39	0.00	0.00	-	-
neos-4387871-tavua	34	34	34	0.00	7.87	-	-
neos-4413714-turia	2016	0	0	0.00	0.00	L	L
neos-4532248-waihi	4972	0	0	0.00	0.00	C	C
neos-4647030-tutaki	799	125	4	0.00	0.00	-	-
neos-4722843-widden	25389	0	0	0.00	0.00	C	C
neos-4738912-atrato	233	0	3	0.00	7.61	L	-
neos-4763324-toguru	1563	0	0	0.00	0.00	C	C

Continued on next page

		# Cuts		Gap Closed (%)		Failure	
Instance	# Frac. Vars.	NX	WX	NX	WX	NX	WX
neos-4954672-berkel	58	58	25	1.12	0.77	-	-
neos-5049753-cuanza	230	0	0	0.00	0.00	C	C
neos-5052403-cygnet	695	2	2	2.79	9.35	-	-
neos-5093327-huahum	64	64	10	1.00	0.86	-	-
neos-5104907-jarama	952	0	0	0.00	0.00	C	C
neos-5107597-kakapo	1519	11	11	0.00	0.00	-	-
neos-5114902-kasavu	259	0	0	0.00	0.00	B	B
neos-5188808-nattai	23	23	23	0.00	0.00	-	-
neos-5195221-niemur	1138	0	0	0.00	0.00	F	F
neos-631710	1139	0	0	0.00	0.00	C	C
neos-662469	348	2	2	4.73	5.30	-	-
neos-787933	22	18	5	0.00	0.00	-	-
neos-827175	0	0	0	0.00	0.00	N	N
neos-860300	106	106	106	9.64	13.04	-	-
neos-873061	92	0	0	0.00	0.00	C	C
neos-911970	55	55	55	0.00	0.00	-	-
neos-933966	1275	0	0	0.00	0.00	F	F
neos-950242	116	2	2	0.00	0.00	-	-
neos-957323	123	0	0	0.00	0.00	F	F
neos-960392	404	2	2	0.00	0.00	-	-
neos17	171	171	171	0.00	0.15	-	-
neos5	35	35	35	23.21	23.21	-	-
neos8	193	193	174	0.00	0.00	-	-
net12	374	71	3	0.00	0.00	-	-
netdiversion	3902	0	0	0.00	0.00	C	C
nexp-150-20-8-5	74	74	74	0.08	0.57	-	-
ns1208400	324	0	0	0.00	0.00	F	F
ns1644855	343	0	0	0.00	0.00	B	B
ns1760995	1060	0	0	0.00	0.00	C	C
ns1830653	167	155	84	12.57	14.41	-	-
nu25-pr12	37	37	37	11.03	12.99	-	-
nursesched-medium-hint03	1224	0	0	0.00	0.00	F	F
nursesched-sprint02	448	397	285	0.00	0.00	-	-
nw04	6	6	6	0.00	0.03	-	-
opm2-z10-s4	5584	0	0	0.00	0.00	B	B
p200x1188c	5	5	5	2.97	8.87	-	-
peg-solitaire-a3	1005	0	0	0.00	0.00	Z	Z
pg	93	93	6	0.00	0.34	-	-
pg5_34	88	88	25	0.00	0.36	-	-
physiciansched3-3	1658	0	0	0.00	0.00	F	F
physiciansched6-2	974	0	0	0.00	0.00	F	F
piperout-08	1595	0	0	0.00	0.00	Z	Z
piperout-27	2480	1114	4	0.00	0.00	-	-
pk1	15	0	0	0.00	0.00	F	F
proteindesign121hz512p9	209	84	86	0.00	0.00	-	-
proteindesign122trx11p8	163	0	0	0.00	0.00	Z	Z
qap10	1218	243	12	0.00	34.74	-	-
radiationm18-12-05	945	472	6	3.12	3.59	-	-
radiationm40-10-02	2964	0	0	0.00	0.00	F	F
rail01	7535	0	0	0.00	0.00	B	B

Continued on next page

		# Cuts		Gap Closed (%)		Failure	
Instance	# Frac. Vars.	NX	WX	NX	WX	NX	WX
rail02	10824	0	0	0.00	0.00	B	B
rail507	254	2	2	3.76	8.16	-	-
ran14x18-disj-8	86	86	86	6.73	7.40	-	-
reblock115	878	72	0	0.00	0.00	-	F
rmatr100-p10	51	51	51	29.06	31.45	-	-
rmatr200-p5	66	0	0	0.00	0.00	F	F
rocI-4-11	421	54	51	0.00	0.00	-	-
rocII-5-11	181	181	181	0.00	0.05	-	-
rococoB10-011000	261	120	261	0.64	0.80	-	-
rococoC10-001000	188	147	188	1.82	3.72	-	-
roi2alpha3n4	37	37	18	0.01	3.48	-	-
roi5alpha10n8	89	73	12	0.00	3.03	-	-
roll3000	200	200	47	0.03	0.24	-	-
s100	198	0	0	0.00	0.00	B	B
s250r10	294	0	0	0.00	0.00	C	C
satellites2-40	1767	0	0	0.00	0.00	F	F
satellites2-60-fs	1704	0	0	0.00	0.00	C	B
savsched1	15794	0	0	0.00	0.00	B	B
sct2	49	0	0	0.00	0.00	F	F
seymour	544	544	438	11.47	11.59	-	-
seymour1	135	135	135	21.62	22.43	-	-
sing326	934	102	0	0.00	0.00	-	F
sing44	760	256	132	0.00	2.08	-	-
snp-02-004-104	68	0	0	0.00	0.00	C	C
sorrell3	1024	84	74	6.35	6.35	-	-
sp150x300d	42	42	42	2.79	2.79	-	-
sp97ar	194	0	0	0.00	0.00	L	L
sp98ar	157	0	0	0.00	0.00	L	F
splice1k1	290	17	11	0.00	0.05	-	-
square41	342	0	0	0.00	0.00	B	B
square47	347	0	0	0.00	0.00	B	B
supportcase10	7705	0	0	0.00	0.00	B	B
supportcase12	190	2	2	0.00	0.00	-	-
supportcase18	100	0	0	0.00	0.00	F	F
supportcase22	62	0	0	0.00	0.00	B	B
supportcase26	236	16	13	0.00	0.00	-	-
supportcase33	269	60	2	0.00	15.39	-	-
supportcase40	38	38	5	17.98	18.78	-	-
supportcase42	151	0	0	0.00	0.00	Z	Z
supportcase6	140	0	0	0.00	0.00	Z	Z
supportcase7	253	72	5	0.00	0.12	-	-
swath1	12	12	12	0.00	0.00	-	-
swath3	16	16	16	6.78	6.60	-	-
thor50dday	54	0	0	0.00	0.00	C	C
timtab1	109	109	45	1.15	1.44	-	-
tr12-30	326	326	326	1.07	1.10	-	-
traininstance2	110	0	0	0.00	0.00	F	F
traininstance6	11	11	11	0.00	0.00	-	-
trento1	482	2	0	0.00	0.00	-	L
triptim1	3454	0	0	0.00	0.00	F	F

Continued on next page

Instance	# Frac. Vars.	# Cuts NX	WX	Gap Closed (%) NX	WX	Failure NX	WX
uccase12	81	0	0	0.00	0.00	C	C
uccase9	422	422	28	0.02	0.07	-	-
uct-subprob	210	127	210	3.76	5.67	-	-
unitcal_7	719	0	9	0.00	0.36	L	-
var-smallemery-m6j6	396	396	396	13.24	11.06	-	-
wachplan	285	0	0	0.00	0.00	F	F

Chapter C

Set B Instance Set

In later experiments, we only consider instances for which at least one VPC cut was generated in the comparison between a single round of Gomory cuts and VPCs. The instances are:

1. `50v-10`
2. `CMS750_4`
3. `air05`
4. `app1-1`
5. `assign1-5-8`
6. `atlanta-ip`
7. `b1c1s1`
8. `beasleyC3`
9. `binkar10_1`
10. `blp-ar98`
11. `blp-ic98`
12. `cod105`
13. `cost266-UUE`
14. `csched007`
15. `dano3_3`
16. `decomp2`
17. `dws008-01`
18. `eil33-2`
19. `exp-1-500-5-5`
20. `fast0507`
21. `fiball`
22. `gen-ip002`

23. gen-ip054
24. glass-sc
25. glass4
26. gmu-35-40
27. gmu-35-50
28. graph20-20-1rand
29. graphdraw-domain
30. h80x6320d
31. hypothyroid-k1
32. ic97_potential
33. icir97_tension
34. irp
35. istanbul-no-cutoff
36. k1mushroom
37. lectsched-5-obj
38. lotsize
39. mas74
40. mas76
41. mc11
42. mcsched
43. mik-250-20-75-4
44. milo-v12-6-r2-40-1
45. mushroom-best
46. mzzv42z
47. n3div36
48. n5-3
49. neos-1445765
50. neos-1456979
51. neos-1582420
52. neos-3024952-loue
53. neos-3046615-murg
54. neos-3381206-awhea
55. neos-3627168-kasai
56. neos-3656078-kumeu

57. neos-3754480-nidda
58. neos-4300652-rahue
59. neos-4338804-snowy
60. neos-4387871-tavua
61. neos-4647030-tutaki
62. neos-4738912-atrato
63. neos-4954672-berkel
64. neos-5052403-cygnet
65. neos-5093327-huahum
66. neos-5107597-kakapo
67. neos-5188808-nattai
68. neos-5195221-niemur
69. neos-662469
70. neos-787933
71. neos-860300
72. neos-911970
73. neos-950242
74. neos-960392
75. neos17
76. neos5
77. neos8
78. net12
79. nexp-150-20-8-5
80. ns1830653
81. nu25-pr12
82. nursesched-sprint02
83. nw04
84. p200x1188c
85. pg
86. pg5_34
87. piperout-27
88. proteindesign121hz512p9
89. qap10
90. radiationm18-12-05

91. `rail507`
92. `ran14x18-disj-8`
93. `rmatr100-p10`
94. `rocI-4-11`
95. `rocII-5-11`
96. `rococoB10-011000`
97. `rococoC10-001000`
98. `roi2alpha3n4`
99. `roi5alpha10n8`
100. `roll3000`
101. `seymour`
102. `seymour1`
103. `sing326`
104. `sing44`
105. `sorrell3`
106. `sp150x300d`
107. `splice1k1`
108. `supportcase12`
109. `supportcase26`
110. `supportcase33`
111. `supportcase7`
112. `swath1`
113. `swath3`
114. `timtab1`
115. `tr12-30`
116. `traininstance6`
117. `uccase9`
118. `uct-subprob`
119. `unitcal_7`
120. `var-smallemery-m6j6`

Chapter D

Branch-and-Bound Experiment Results

Table D.1: Branch-and-bound experiment results for the default configuration. Columns r1, r2, and r3 refer to runs 1, 2, and 3 respectively. Columns labeled gmean show the shifted geometric mean over the three runs.

	Final Gap (%)				#Cuts			Solve Time (s)				Node Count				Time/Node (s)			
instance	r1	r2	r3	gmean	r1	r2	r3	r1	r2	r3	gmean	r1	r2	r3	gmean	r1	r2	r3	gmean
50v-10	0.01	0.01	0.01	0.01	0	0	0	7200.02	7200.04	7200.01	7200.02	103171.00	88990.00	124185.00	104469.46	0.07	0.08	0.06	0.07
CMS750_4	0.00	0.00	0.00	0.00	0	0	0	527.15	898.93	1100.94	805.07	2073.00	0.00	4913.00	215.81	0.25	inf	0.22	inf
air05	0.00	0.00	0.00	0.00	0	0	0	18.77	23.58	22.19	21.42	13.00	65.00	33.00	30.55	1.44	0.36	0.67	0.77
app1-1	0.00	0.00	0.00	0.00	0	0	0	44.86	55.06	39.83	46.17	1.00	1.00	1.00	1.00	44.86	55.06	39.83	46.17
assign1-5-8	0.05	0.05	0.05	0.05	0	0	0	7200.01	7200.01	7200.02	7200.01	1272143.00	1418532.00	1468746.00	1383908.12	0.01	0.01	0.00	0.01
atlanta-ip	0.00	0.00	0.00	0.00	0	0	0	6625.22	6913.57	5377.56	6268.49	5883.00	4243.00	3796.00	4558.98	1.13	1.63	1.42	1.38
b1c1s1	0.13	0.13	0.13	0.13	0	0	0	7200.01	7200.02	7200.00	7200.01	98925.00	92495.00	96768.00	96025.23	0.07	0.08	0.07	0.08
beasleyC3	0.00	0.00	0.00	0.00	0	0	0	30.47	23.90	23.97	25.95	1.00	1.00	1.00	1.00	30.47	23.90	23.97	25.95
binkar10_1	0.00	0.00	0.00	0.00	0	0	0	70.20	70.03	70.73	70.32	2073.00	2626.00	1911.00	2183.00	0.03	0.03	0.04	0.03
blp-ar98	0.00	0.00	0.00	0.00	0	0	0	7200.06	6401.31	7200.00	6923.29	64305.00	66764.00	74839.00	68491.77	0.11	0.10	0.10	0.10
blp-ic98	0.00	0.00	0.00	0.00	0	0	0	5239.33	2462.46	2497.61	3182.23	82154.00	34342.00	21042.00	39010.56	0.06	0.07	0.12	0.08
cod105	0.00	0.34	0.00	0.10	0	0	0	310.37	7200.03	809.21	1219.17	1411.00	59547.00	5481.00	7723.68	0.22	0.12	0.15	0.16
cost266-UUE	0.02	0.01	0.03	0.02	0	0	0	7200.00	7200.00	7200.01	7200.00	209519.00	204215.00	190521.00	201257.14	0.03	0.04	0.04	0.04
csched007	0.03	0.00	0.00	0.01	0	0	0	7200.01	3339.77	4362.67	4716.34	129455.00	54414.00	75352.00	80967.07	0.06	0.06	0.06	0.06
dano3_3	0.00	0.00	0.00	0.00	0	0	0	167.98	127.66	171.32	154.31	11.00	16.00	13.00	13.19	15.27	7.98	13.18	11.75
decomp2	0.00	0.00	0.00	0.00	0	0	0	0.08	0.11	0.11	0.10	0.00	0.00	0.00	0.00	inf	inf	inf	inf
dws008-01	0.40	0.41	0.40	0.40	0	0	0	7200.02	7200.00	7200.03	7200.02	34162.00	38526.00	35835.00	36130.02	0.21	0.19	0.20	0.20
eil33-2	0.00	0.00	0.00	0.00	0	0	0	176.02	155.89	125.78	151.13	319.00	435.00	351.00	365.21	0.55	0.36	0.36	0.42
exp-1-500-5-5	0.00	0.00	0.00	0.00	0	0	0	3.53	2.56	3.57	3.19	1.00	1.00	1.00	1.00	3.53	2.56	3.57	3.19
fast0507	0.00	0.00	0.00	0.00	0	0	0	123.77	107.52	98.05	109.28	497.00	527.00	481.00	501.31	0.25	0.20	0.20	0.22
fiball	0.00	0.00	0.00	0.00	0	0	0	4.71	3.44	4.32	4.13	1.00	1.00	1.00	1.00	4.71	3.44	4.32	4.13
gen-ip002	0.00	0.00	0.00	0.00	0	0	0	3717.21	3865.99	4253.19	3939.11	4531456.00	4631519.00	4992942.00	4714539.15	0.00	0.00	0.00	0.00
gen-ip054	0.00	0.00	0.00	0.00	0	0	0	4564.56	4496.53	3534.52	4170.64	5297636.00	6066866.00	4413192.00	5215145.71	0.00	0.00	0.00	0.00
glass-sc	0.12	0.12	0.09	0.11	0	0	0	7200.02	7200.01	7200.01	7200.01	179089.00	164005.00	222316.00	186909.88	0.04	0.04	0.03	0.04
glass4	0.00	0.00	0.00	0.00	0	0	0	2041.08	4607.44	1239.95	2267.79	526476.00	923741.00	214897.00	471035.06	0.00	0.00	0.01	0.00
gmu-35-40	0.00	0.00	0.00	0.00	0	0	0	7200.03	7200.02	7200.01	7200.02	1803471.00	1883633.00	2105982.00	1926874.24	0.00	0.00	0.00	0.00
gmu-35-50	0.00	0.00	0.00	0.00	0	0	0	7200.01	7200.02	7200.13	7200.05	789407.00	878544.00	885461.00	849985.09	0.01	0.01	0.01	0.01
graph20-20-1rand	0.00	0.43	0.51	0.29	0	0	0	5340.47	7200.00	7200.02	6517.52	104724.00	177617.00	184061.00	150717.71	0.05	0.04	0.04	0.04
graphdraw-domain	0.00	0.00	0.00	0.00	0	0	0	4848.98	2631.26	3728.34	3623.38	1029926.00	557405.00	874747.00	794852.36	0.00	0.00	0.00	0.00
h80x6320d	0.00	0.00	0.00	0.00	0	0	0	201.71	211.79	226.41	213.06	9.00	5.00	1.00	3.93	22.41	42.36	226.41	60.34
hypothyroid-k1	0.00	0.00	0.00	0.00	0	0	0	44.47	36.25	39.36	39.89	0.00	0.00	0.00	0.00	inf	inf	inf	inf
ic97_potential	0.00	0.01	0.00	0.00	0	0	0	7200.01	7200.01	7200.00	7200.01	957192.00	1037431.00	881763.00	956687.40	0.01	0.01	0.01	0.01
icir97_tension	0.00	0.00	0.00	0.00	0	0	0	4891.19	7200.01	7200.01	6329.38	187744.00	230733.00	239772.00	218184.93	0.03	0.03	0.03	0.03
irp	0.00	0.00	0.00	0.00	0	0	0	14.42	13.52	14.12	14.02	1.00	1.00	1.00	1.00	14.42	13.52	14.12	14.02
istanbul-no-cutoff	0.00	0.00	0.00	0.00	0	0	0	167.01	197.55	178.92	180.73	431.00	450.00	281.00	379.16	0.39	0.44	0.64	0.48
k1mushroom	0.00	0.00	0.00	0.00	0	0	0	2061.17	2618.06	2595.27	2410.42	0.00	0.00	0.00	0.00	inf	inf	inf	inf

Continued on next page

Table D.1: Branch-and-bound experiment results for the default configuration. Columns r1, r2, and r3 refer to runs 1, 2, and 3 respectively. Cut Time denotes the cut generation time. Columns labeled gmean show the shifted geometric mean over the three runs.

	Final Gap (%)				#Cuts			Solve Time (s)				Node Count				Time/Node (s)			
instance	r1	r2	r3	gmean	r1	r2	r3	r1	r2	r3	gmean	r1	r2	r3	gmean	r1	r2	r3	gmean
lectsched-5-obj	0.60	0.60	0.60	0.60	0	0	0	7200.02	7200.01	7200.05	7200.03	66076.00	60302.00	73246.00	66331.50	0.11	0.12	0.10	0.11
lotsize	0.01	0.01	0.12	0.05	0	0	0	3001.08	1389.18	7200.01	3108.06	3159.00	135.00	27937.00	2288.85	0.95	10.29	0.26	2.03
mas74	0.00	0.00	0.00	0.00	0	0	0	4595.02	4267.62	3976.59	4272.30	4920799.00	5002610.00	4657875.00	4858177.38	0.00	0.00	0.00	0.00
mas76	0.00	0.00	0.00	0.00	0	0	0	311.02	360.67	350.98	340.20	307390.00	307905.00	307034.00	307442.79	0.00	0.00	0.00	0.00
mc11	0.00	0.00	0.00	0.00	0	0	0	310.70	225.93	187.80	236.25	1241.00	608.00	411.00	676.97	0.25	0.37	0.46	0.36
mcsched	0.00	0.00	0.00	0.00	0	0	0	357.30	516.30	399.29	419.20	4315.00	4607.00	5498.00	4781.18	0.08	0.11	0.07	0.09
mik-250-20-75-4	0.00	0.00	0.00	0.00	0	0	0	166.77	286.61	115.77	176.95	6528.00	12825.00	6606.00	8208.45	0.03	0.02	0.02	0.02
milo-v12-6-r2-40-1	0.07	0.07	0.04	0.06	0	0	0	7200.01	7200.01	7200.00	7200.01	47844.00	62748.00	46126.00	51735.80	0.15	0.11	0.16	0.14
mushroom-best	0.00	0.00	0.00	0.00	0	0	0	4867.74	4924.49	6270.80	5317.05	17496.00	18864.00	21104.00	19097.62	0.28	0.26	0.30	0.28
mzzv42z	0.00	0.00	0.00	0.00	0	0	0	195.75	197.61	200.81	198.04	1.00	1.00	1.00	1.00	195.75	197.61	200.81	198.05
n3div36	0.03	0.03	0.03	0.03	0	0	0	7200.00	7200.11	7200.01	7200.04	85877.00	79874.00	80088.00	81899.88	0.08	0.09	0.09	0.09
n5-3	0.00	0.00	0.00	0.00	0	0	0	63.69	60.28	50.59	57.92	532.00	607.00	678.00	602.72	0.12	0.10	0.07	0.10
neos-1445765	0.00	0.00	0.00	0.00	0	0	0	61.94	88.15	79.77	75.81	1.00	1.00	1.00	1.00	61.94	88.15	79.77	75.81
neos-1456979	0.00	0.00	0.00	0.00	0	0	0	371.60	330.70	330.02	343.57	2299.00	2375.00	1708.00	2104.91	0.16	0.14	0.19	0.16
neos-1582420	0.00	0.00	0.00	0.00	0	0	0	10.15	3.85	7.03	6.57	1.00	1.00	1.00	1.00	10.15	3.85	7.03	6.57
neos-3024952-loue	0.00	0.00	0.00	0.00	0	0	0	7200.01	7200.00	7200.01	7200.01	13072.00	23344.00	32492.00	21483.22	0.55	0.31	0.22	0.35
neos-3046615-murg	1.87	2.00	1.86	1.91	0	0	0	7200.05	7200.05	7200.05	7200.05	1696811.00	1387980.00	1980712.00	1670882.87	0.00	0.01	0.00	0.00
neos-3381206-awhea	0.00	0.00	0.00	0.00	0	0	0	5.10	1.60	3.71	3.21	37.00	1.00	21.00	10.87	0.14	1.60	0.18	0.52
neos-3627168-kasai	0.00	0.00	0.00	0.00	0	0	0	595.19	2170.65	1076.58	1116.40	26663.00	124548.00	43398.00	52429.23	0.02	0.02	0.02	0.02
neos-3656078-kumeu	0.37	0.11	0.09	0.18	0	0	0	7200.07	7200.16	7200.08	7200.10	1850.00	6361.00	4270.00	3690.28	3.89	1.13	1.69	2.04
neos-3754480-nidda	Inf	Inf	Inf	Inf	0	0	0	7200.04	7200.04	7200.01	7200.03	4177111.00	4154891.00	4146039.00	4159326.48	0.00	0.00	0.00	0.00
neos-4300652-rahue	1.86	2.84	2.04	2.22	0	0	0	7200.19	7200.14	7200.21	7200.18	1669.00	1282.00	1555.00	1492.89	4.31	5.62	4.63	4.83
neos-4338804-snowy	0.02	0.02	0.02	0.02	0	0	0	7200.02	7200.01	7200.01	7200.01	970481.00	1081391.00	1207627.00	1082183.40	0.01	0.01	0.01	0.01
neos-4387871-tavua	0.14	0.16	0.14	0.15	0	0	0	7200.13	7200.03	7200.10	7200.09	5419.00	6256.00	7453.00	6321.93	1.33	1.15	0.97	1.14
neos-4647030-tutaki	0.00	0.00	0.00	0.00	0	0	0	7200.25	7200.13	7200.13	7200.17	2243.00	3836.00	5070.00	3520.30	3.21	1.88	1.42	2.08
neos-4738912-atrato	0.00	0.00	0.00	0.00	0	0	0	1673.07	1152.65	849.71	1178.98	12717.00	7931.00	5982.00	8449.98	0.13	0.15	0.14	0.14
neos-4954672-berkel	0.09	0.08	0.11	0.09	0	0	0	7200.00	7200.01	7200.01	7200.01	216413.00	180186.00	193633.00	196182.95	0.03	0.04	0.04	0.04
neos-5052403-cygnet	0.01	0.01	0.01	0.01	0	0	0	7200.18	7199.11	7199.48	7199.59	1445.00	143.00	21.00	165.08	4.98	50.34	342.83	46.27
neos-5093327-huahum	0.10	0.12	0.12	0.11	0	0	0	7200.06	7200.15	7200.04	7200.08	12711.00	8313.00	8462.00	9633.95	0.57	0.87	0.85	0.76
neos-5107597-kakapo	0.78	1.13	1.11	1.00	0	0	0	7200.00	7200.03	7200.01	7200.01	242644.00	225865.00	220249.00	229391.58	0.03	0.03	0.03	0.03
neos-5188808-nattai	0.00	0.00	0.00	0.00	0	0	0	2576.79	2775.41	3318.84	2873.85	9296.00	11587.00	14866.00	11699.15	0.28	0.24	0.22	0.25
neos-5195221-niemur	0.00	0.00	0.00	0.00	0	0	0	4162.19	3002.30	3476.44	3515.37	29841.00	21225.00	30260.00	26761.49	0.14	0.14	0.11	0.13
neos-662469	0.00	0.00	0.00	0.00	0	0	0	278.95	200.27	200.74	223.85	1053.00	1071.00	1017.00	1046.76	0.26	0.19	0.20	0.22
neos-787933	0.00	0.00	0.00	0.00	0	0	0	2.29	2.74	2.52	2.51	1.00	1.00	1.00	1.00	2.29	2.74	2.52	2.51
neos-860300	0.00	0.00	0.00	0.00	0	0	0	18.52	17.23	12.23	15.76	1.00	1.00	1.00	1.00	18.52	17.23	12.23	15.76
neos-911970	0.00	0.00	0.00	0.00	0	0	0	38.60	18.10	26.83	26.61	187.00	449.00	85.00	192.77	0.21	0.04	0.32	0.18

Continued on next page

Table D.1: Branch-and-bound experiment results for the default configuration. Columns r1, r2, and r3 refer to runs 1, 2, and 3 respectively. Cut Time denotes the cut generation time. Columns labeled gmean show the shifted geometric mean over the three runs.

	Final Gap (%)				#Cuts			Solve Time (s)				Node Count				Time/Node (s)			
instance	r1	r2	r3	gmean	r1	r2	r3	r1	r2	r3	gmean	r1	r2	r3	gmean	r1	r2	r3	gmean
neos-950242	0.00	0.00	0.00	0.00	0	0	0	114.05	96.36	306.47	150.02	46.00	38.00	76.00	51.07	2.48	2.54	4.03	2.96
neos-960392	0.00	0.00	0.00	0.00	0	0	0	92.55	40.44	116.29	75.90	1.00	1.00	1.00	1.00	92.55	40.44	116.29	75.90
neos17	0.00	0.00	0.00	0.00	0	0	0	26.68	47.01	25.84	31.92	2093.00	4351.00	2327.00	2767.32	0.01	0.01	0.01	0.01
neos5	0.00	0.00	0.00	0.00	0	0	0	4336.01	1972.77	1978.84	2567.66	2526824.00	977113.00	1125626.00	1405954.69	0.00	0.00	0.00	0.00
neos8	0.00	0.00	0.00	0.00	0	0	0	3.60	3.88	2.72	3.37	1.00	1.00	1.00	1.00	3.60	3.88	2.72	3.37
net12	0.00	0.00	0.00	0.00	0	0	0	229.75	307.93	302.88	277.76	764.00	1012.00	923.00	893.64	0.30	0.30	0.33	0.31
nexp-150-20-8-5	0.00	0.00	0.00	0.00	0	0	0	509.98	462.78	1005.66	619.20	1.00	1.00	1.00	1.00	509.98	462.78	1005.66	619.20
ns1830653	0.00	0.00	0.00	0.00	0	0	0	237.30	294.15	310.42	278.80	3141.00	3157.00	3851.00	3367.49	0.08	0.09	0.08	0.08
nu25-pr12	0.00	0.00	0.00	0.00	0	0	0	13.68	14.03	10.56	12.66	9.00	1.00	1.00	2.42	1.52	14.03	10.56	6.59
nursesched-sprint02	0.00	0.00	0.00	0.00	0	0	0	3.51	5.42	4.80	4.52	1.00	1.00	1.00	1.00	3.51	5.42	4.80	4.52
nw04	0.00	0.00	0.00	0.00	0	0	0	34.81	39.39	35.86	36.64	1.00	1.00	1.00	1.00	34.81	39.39	35.86	36.64
p200x1188c	0.00	0.00	0.00	0.00	0	0	0	13.05	8.05	9.12	9.87	1.00	1.00	1.00	1.00	13.05	8.05	9.12	9.88
pg	0.00	0.00	0.00	0.00	0	0	0	30.24	28.82	32.38	30.44	159.00	135.00	139.00	143.96	0.19	0.21	0.23	0.21
pg5_34	0.00	0.00	0.00	0.00	0	0	0	2519.00	1754.27	2020.89	2074.71	92789.00	79519.00	83917.00	85232.61	0.03	0.02	0.02	0.02
piperout-27	0.00	0.00	0.00	0.00	0	0	0	33.34	26.61	33.65	31.03	1.00	1.00	1.00	1.00	33.34	26.61	33.65	31.03
proteindesign121hz512p9	0.02	0.02	0.02	0.02	0	0	0	7200.39	7200.38	7199.44	7200.07	1.00	1.00	430.00	10.99	7200.39	7200.38	16.74	971.64
qap10	0.00	0.00	0.00	0.00	0	0	0	57.45	47.75	75.71	59.24	1.00	1.00	1.00	1.00	57.45	47.75	75.71	59.24
radiationm18-12-05	0.00	0.00	0.13	0.04	0	0	0	7200.00	7200.03	7200.06	7200.03	198241.00	161137.00	158348.00	171659.22	0.04	0.04	0.05	0.04
rail507	0.00	0.00	0.00	0.00	0	0	0	104.17	100.17	91.15	98.34	469.00	469.00	371.00	433.76	0.22	0.21	0.25	0.23
ran14x18-disj-8	0.00	0.00	0.00	0.00	0	0	0	3557.32	4285.55	2979.44	3567.98	140659.00	198695.00	140658.00	157823.98	0.03	0.02	0.02	0.02
rmatr100-p10	0.00	0.00	0.00	0.00	0	0	0	178.43	187.13	190.86	185.40	757.00	815.00	697.00	754.80	0.24	0.23	0.27	0.25
rocI-4-11	0.00	0.00	0.00	0.00	0	0	0	246.87	182.40	335.04	247.13	17003.00	10701.00	20883.00	15604.46	0.01	0.02	0.02	0.02
rocII-5-11	0.76	0.77	0.77	0.77	0	0	0	7200.01	7200.00	7200.00	7200.00	166668.00	141822.00	138173.00	148367.81	0.04	0.05	0.05	0.05
rococoB10-011000	0.14	0.19	0.14	0.16	0	0	0	7200.01	7200.00	7200.00	7200.00	161818.00	133367.00	135274.00	142921.16	0.04	0.05	0.05	0.05
rococoC10-001000	0.05	0.05	0.02	0.04	0	0	0	7200.00	7200.00	7200.02	7200.01	199974.00	167145.00	174636.00	180052.95	0.04	0.04	0.04	0.04
roi2alpha3n4	0.00	0.00	0.00	0.00	0	0	0	1926.28	1957.27	2627.53	2147.71	3816.00	3239.00	4575.00	3838.29	0.50	0.60	0.57	0.56
roi5alpha10n8	0.31	0.32	0.32	0.32	0	0	0	7200.12	7200.00	7200.01	7200.04	236.00	1610.00	1105.00	749.24	30.51	4.47	6.52	9.90
roll3000	0.00	0.00	0.00	0.00	0	0	0	235.98	109.85	181.93	167.75	1752.00	505.00	1406.00	1075.65	0.13	0.22	0.13	0.16
seymour	0.01	0.01	0.01	0.01	0	0	0	7200.00	7200.02	7200.03	7200.02	68416.00	68216.00	60370.00	65557.40	0.11	0.11	0.12	0.11
seymour1	0.00	0.00	0.00	0.00	0	0	0	108.90	105.40	138.30	116.66	877.00	677.00	885.00	806.95	0.12	0.16	0.16	0.15
sing326	0.00	0.00	0.00	0.00	0	0	0	7200.09	7200.07	7200.09	7200.08	1143.00	1.00	261.00	83.32	6.30	7200.07	27.59	113.54
sing44	0.00	0.00	0.00	0.00	0	0	0	7200.26	7199.68	7200.09	7200.01	3349.00	1932.00	2410.00	2498.34	2.15	3.73	2.99	2.90
sorrell3	0.17	0.17	0.18	0.17	0	0	0	7200.08	7200.10	7200.03	7200.07	2321.00	2278.00	1803.00	2120.36	3.10	3.16	3.99	3.40
sp150x300d	0.00	0.00	0.00	0.00	0	0	0	3.12	32.73	46.11	17.71	230.00	3949.00	6814.00	1837.90	0.01	0.01	0.01	0.01
splice1k1	3.02	3.02	3.03	3.02	0	0	0	7200.00	7200.00	7200.00	7200.00	1255.00	1809.00	1940.00	1639.20	5.74	3.98	3.71	4.41
supportcase12	0.01	0.00	0.00	0.00	0	0	0	7200.00	7200.00	7200.01	7200.01	15584.00	14645.00	13849.00	14675.59	0.46	0.49	0.52	0.49

Continued on next page

Table D.1: Branch-and-bound experiment results for the default configuration. Columns r1, r2, and r3 refer to runs 1, 2, and 3 respectively. Cut Time denotes the cut generation time. Columns labeled gmean show the shifted geometric mean over the three runs.

	Final Gap (%)				#Cuts			Solve Time (s)				Node Count				Time/Node (s)			
instance	r1	r2	r3	gmean	r1	r2	r3	r1	r2	r3	gmean	r1	r2	r3	gmean	r1	r2	r3	gmean
supportcase26	0.10	0.06	0.08	0.08	0	0	0	7200.01	7200.00	7200.01	7200.01	874855.00	961111.00	1028580.00	952757.92	0.01	0.01	0.01	0.01
supportcase33	0.00	0.18	0.00	0.06	0	0	0	1057.47	7200.00	898.55	1898.76	12301.00	67655.00	9509.00	19928.09	0.09	0.11	0.09	0.10
supportcase7	0.00	0.00	0.00	0.00	0	0	0	203.05	227.78	219.71	216.60	7.00	7.00	5.00	6.27	29.01	32.54	43.94	34.63
swath1	0.00	0.00	0.00	0.00	0	0	0	68.71	61.83	77.18	68.96	198.00	245.00	234.00	224.75	0.35	0.25	0.33	0.31
swath3	0.00	0.00	0.00	0.00	0	0	0	1807.63	1133.90	2169.34	1644.43	37487.00	19586.00	36502.00	29925.98	0.05	0.06	0.06	0.06
timtab1	0.16	0.00	0.00	0.05	0	0	0	232.31	469.26	228.85	292.25	11403.00	37625.00	17884.00	19723.75	0.02	0.01	0.01	0.02
tr12-30	0.00	0.00	0.00	0.00	0	0	0	2487.84	2464.31	2798.37	2579.14	182523.00	180011.00	207699.00	189678.03	0.01	0.01	0.01	0.01
traininstance6	5.58	1.13	2.08	2.51	0	0	0	7200.00	7200.00	7200.01	7200.00	258582.00	484391.00	867038.00	477101.68	0.03	0.01	0.01	0.02
uccase9	0.01	0.01	0.01	0.01	0	0	0	7200.22	7202.17	7200.24	7200.88	893.00	1248.00	1033.00	1048.08	8.06	5.77	6.97	6.88
uct-subprob	0.00	0.00	0.00	0.00	0	0	0	3545.17	2591.97	4040.79	3336.16	56371.00	27951.00	66380.00	47115.54	0.06	0.09	0.06	0.07
unitcal_7	0.00	0.00	0.00	0.00	0	0	0	520.98	499.38	471.62	496.92	129.00	186.00	119.00	141.89	4.04	2.68	3.96	3.52
var-smallemery-m6j6	0.01	0.01	0.01	0.01	0	0	0	7200.02	7200.00	7200.01	7200.01	222089.00	207934.00	161233.00	195270.00	0.03	0.03	0.04	0.04

Table D.2: Branch-and-bound experiment results for VPC applied without the restart requirement. Columns r1, r2, and r3 refer to runs 1, 2, and 3 respectively. Cut Time denotes the cut generation time. Columns labeled gmean show the shifted geometric mean over the three runs.

	Final Gap (%)				#Cuts			Solve Time (s)				Node Count				Cut Time (s)			Time/Node (s)			
instance	r1	r2	r3	gmean	r1	r2	r3	r1	r2	r3	gmean	r1	r2	r3	gmean	r1	r2	r3	r1	r2	r3	gmean
50v-10	0.01	0.01	0.01	0.01	172	135	98	7200.00	7200.00	7200.03	7200.01	69046.00	61945.00	79378.00	69761.37	926.43	922.89	926.76	0.10	0.12	0.09	0.10
CMS750_4	0.00	0.00	0.00	0.00	0	0	0	703.29	1980.03	1237.14	1198.88	297.00	3234.00	6811.00	1871.64	0.00	0.00	0.00	2.37	0.61	0.18	0.86
air05	0.00	0.00	0.00	0.00	146	33	138	941.98	464.71	942.83	744.60	9.00	9.00	11.00	9.63	919.85	442.75	920.65	104.66	51.63	85.71	77.42
app1-1	0.00	0.00	0.00	0.00	0	0	4	63.02	51.63	69.94	61.06	1.00	1.00	1.00	1.00	0.00	0.00	55.18	63.02	51.63	69.94	61.06
assign1-5-8	0.06	0.06	0.05	0.06	200	200	200	7200.01	7200.01	7200.01	7200.01	1161195.00	1107850.00	1099004.00	1122350.40	41.50	53.71	34.60	0.01	0.01	0.01	0.01
atlanta-ip	0.00	0.00	0.00	0.00	0	87	22	5512.98	7127.74	5927.07	6152.62	2924.00	2705.00	3416.00	3000.69	0.00	1250.82	881.90	1.89	2.64	1.74	2.06
b1c1s1	0.14	0.13	0.14	0.13	0	0	0	7200.01	7200.01	7200.07	7200.03	97715.00	80091.00	78402.00	84974.91	0.00	0.00	0.00	0.07	0.09	0.09	0.09
beasleyC3	0.00	0.00	0.00	0.00	200	81	0	923.06	953.80	25.10	283.50	1.00	1.00	1.00	1.00	898.54	930.69	0.00	923.06	953.80	25.10	283.50
binkar10_1	0.00	0.00	0.00	0.00	200	200	200	167.31	218.36	194.36	192.21	1609.00	2468.00	2083.00	2022.40	87.20	89.85	79.89	0.10	0.09	0.09	0.10
blp-ar98	0.00	0.00	0.00	0.00	122	160	117	7200.08	7200.01	7200.04	7200.04	50705.00	52603.00	41655.00	48073.92	971.31	962.75	980.73	0.14	0.14	0.17	0.15
blp-ic98	0.00	0.00	0.00	0.00	190	185	149	5329.08	4741.19	3919.97	4626.74	50091.00	55698.00	33876.00	45550.94	949.53	947.77	952.06	0.11	0.09	0.12	0.10
cod105	0.00	0.35	0.00	0.10	0	0	0	920.12	7200.18	1325.97	2063.72	1461.00	39847.00	5371.00	6788.38	0.00	0.00	0.00	0.63	0.18	0.25	0.34
cost266-UUE	0.02	0.00	0.03	0.02	0	0	0	7200.01	5203.75	7200.00	6461.42	208947.00	138444.00	173177.00	171106.64	0.00	0.00	0.00	0.03	0.04	0.04	0.04
csched007	0.05	0.06	0.02	0.04	200	200	200	7200.01	7200.02	7200.16	7200.06	107651.00	108640.00	98133.00	104698.62	730.53	819.84	792.73	0.07	0.07	0.07	0.07
dano3_3	0.00	0.00	0.00	0.00	81	14	61	235.89	133.89	214.41	189.23	10.00	3.00	16.00	8.08	88.27	33.97	57.78	23.59	44.63	13.40	24.28
decomp2	0.00	0.00	0.00	0.00	0	0	0	0.12	0.09	0.10	0.11	0.00	0.00	0.00	0.00	0.00	0.00	0.00	inf	inf	inf	inf
dws008-01	0.50	0.50	0.44	0.48	62	200	123	7200.02	7200.01	7200.03	7200.02	20936.00	21185.00	26705.00	22794.94	360.76	283.03	470.40	0.34	0.34	0.27	0.32
eil33-2	0.00	0.00	0.00	0.00	200	200	200	457.20	453.03	448.54	452.91	321.00	331.00	391.00	346.34	233.54	185.84	217.01	1.42	1.37	1.15	1.31
exp-1-500-5-5	0.00	0.00	0.00	0.00	0	0	0	3.50	2.57	3.76	3.25	1.00	1.00	1.00	1.00	0.00	0.00	0.00	3.50	2.57	3.76	3.24
fast0507	0.00	0.00	0.00	0.00	3	3	3	417.79	442.41	427.16	429.00	313.00	507.00	391.00	395.91	301.73	331.24	317.04	1.33	0.87	1.09	1.09
fiball	0.00	0.00	0.00	0.00	0	0	0	4.45	3.43	3.48	3.76	1.00	1.00	1.00	1.00	0.00	0.00	0.00	4.45	3.43	3.48	3.76
gen-ip002	0.00	0.00	0.00	0.00	200	200	200	4257.81	4078.97	4178.60	4171.15	4705755.00	4624358.00	4602518.00	4643998.66	5.49	5.55	5.49	0.00	0.00	0.00	0.00
gen-ip054	0.00	0.00	0.00	0.00	200	200	200	6714.11	7200.00	4679.12	6092.98	7962139.00	7048531.00	6458487.00	7129948.40	3.87	3.74	3.21	0.00	0.00	0.00	0.00
glass-sc	0.14	0.12	0.13	0.13	200	200	200	7200.05	7200.05	7200.00	7200.03	182151.00	208391.00	212272.00	200478.47	209.39	221.63	208.67	0.04	0.03	0.03	0.04
glass4	0.00	0.00	0.00	0.00	0	0	200	1135.28	643.33	1197.08	956.25	251083.00	123425.00	211145.00	187039.73	0.00	0.00	58.52	0.00	0.01	0.01	0.01
gmu-35-40	0.00	0.00	0.00	0.00	0	0	200	7200.04	6696.80	7200.03	7028.23	1577813.00	3912762.00	1340932.00	2022935.00	0.00	0.00	15.49	0.00	0.00	0.01	0.00
gmu-35-50	0.00	0.00	0.00	0.00	200	200	200	7200.02	7200.02	7200.02	7200.02	753739.00	735772.00	1059955.00	837691.09	13.90	16.56	12.31	0.01	0.01	0.01	0.01
graph20-20-1rand	0.00	0.55	0.67	0.37	0	138	0	5045.31	7200.00	7200.00	6395.17	123339.00	129429.00	153664.00	134865.67	0.00	938.80	0.00	0.04	0.06	0.05	0.05
graphdraw-domain	0.05	0.00	0.00	0.02	0	200	200	7200.00	3615.12	3344.47	4432.00	1471336.00	854411.00	711847.00	963655.50	0.00	8.94	12.01	0.00	0.00	0.00	0.00
h80x6320d	0.00	0.00	0.00	0.00	0	66	24	634.75	1488.83	1015.70	986.50	5.00	1.00	3.00	2.63	0.00	1235.33	795.03	126.95	1488.83	338.57	400.51
hypothyroid-k1	0.00	0.00	0.00	0.00	0	0	0	39.45	35.79	36.77	37.30	0.00	0.00	0.00	0.00	0.00	0.00	0.00	inf	inf	inf	inf
ic97_potential	0.00	0.01	0.00	0.00	0	0	0	7200.00	7200.00	7200.00	7200.00	823674.00	752552.00	862113.00	811494.08	0.00	0.00	0.00	0.01	0.01	0.01	0.01
icir97_tension	0.00	0.00	0.00	0.00	0	0	0	7200.02	7200.02	7200.00	7200.01	243828.00	235659.00	201319.00	226160.87	0.00	0.00	0.00	0.03	0.03	0.04	0.03
irp	0.00	0.00	0.00	0.00	0	17	0	13.99	45.93	18.85	23.08	1.00	1.00	1.00	1.00	0.00	27.74	0.00	13.99	45.93	18.85	23.08
istanbul-no-cutoff	0.00	0.00	0.00	0.00	14	3	3	444.30	605.21	468.60	501.35	324.00	392.00	293.00	333.87	260.39	387.11	308.19	1.37	1.54	1.60	1.50
k1mushroom	0.00	0.00	0.00	0.00	0	0	0	2085.89	2687.22	2739.38	2485.52	0.00	0.00	0.00	0.00	0.00	0.00	0.00	inf	inf	inf	inf
lectsched-5-obj	0.57	0.60	0.60	0.59	69	69	0	7200.05	7200.04	7200.01	7200.04	86601.00	54123.00	73289.00	70034.92	249.37	115.29	0.00	0.08	0.13	0.10	0.10
lotsize	0.01	0.01	0.02	0.01	0	0	0	7200.02	7200.01	2600.54	5127.69	16177.00	11635.00	1652.00	6775.42	0.00	0.00	0.00	0.45	0.62	1.57	0.82
mas74	0.00	0.00	0.00	0.00	200	200	200	5996.24	3557.08	6556.21	5190.53	5293102.00	3185189.00	5211719.00	4445720.52	8.90	7.76	7.79	0.00	0.00	0.00	0.00
mas76	0.00	0.00	0.00	0.00	200	0	0	469.69	319.38	469.95	413.12	308470.00	317810.00	367748.00	330352.00	7.29	0.00	0.00	0.00	0.00	0.00	0.00
mc11	0.00	0.00	0.00	0.00	200	200	0	724.51	1323.91	471.67	767.77	774.00	3944.00	3.00	229.39	495.56	732.80	0.00	0.94	0.34	157.22	6.42
mcsched	0.00	0.00	0.00	0.00	200	200	200	883.09	887.30	602.23	778.55	13322.00	5460.00	5635.00	7428.31	258.43	237.92	192.33	0.07	0.16	0.11	0.11
mik-250-20-75-4	0.00	0.00	0.00	0.00	200	200	200	209.74	252.99	238.09	232.91	9266.00	13508.00	11847.00	11403.31	13.64	13.29	10.38	0.02	0.02	0.02	0.02
milo-v12-6-r2-40-1	0.06	0.07	0.06	0.06	0	0	0	7200.06	7200.00	7200.02	7200.03	46083.00	49175.00	54591.00	49827.46	0.00	0.00	0.00	0.16	0.15	0.13	0.14
mushroom-best	0.52	0.57	0.47	0.52	200	200	200	7200.03	7200.02	7200.13	7200.06	16266.00	12024.00	17967.00	15203.22	183.93	163.52	170.22	0.44	0.60	0.40	0.48
mzzv42z	0.00	0.00	0.00	0.00	0	0	0	194.64	179.54	201.66	191.72	1.00	1.00	1.00	1.00	0.00	0.00	0.00	194.64	179.54	201.66	191.72
n3div36	0.04	0.04	0.04	0.04	121	99	82	7200.01	7200.07	7200.01	7200.03	52252.00	48975.00	56840.00	52590.89	984.94	980.09	986.07	0.14	0.15	0.13	0.14
n5-3	0.00	0.00	0.00	0.00	23	0	48	657.35	217.32	1059.62	533.20	850.00	780.00	1338.00	960.88	572.53	0.00	923.77	0.77	0.28	0.79	0.60
neos-1445765	0.00	0.00	0.00	0.00	200	200	200	816.57	760.64	793.37	789.86	37.00	1.00	1.00	4.34	741.47	677.82	704.71	22.07	760.64	793.37	239.77
neos-1456979	0.00	0.00	0.00	0.00	200	200	200	806.44	715.43	1393.79	929.96	4151.00	2401.00	3599.00	3297.96	420.98	413.67	635.89	0.19	0.30	0.39	0.29
neos-1582420	0.00	0.00	0.00	0.00	0	0	0	16.44	2.93	4.66	6.29	1.00	1.00	1.00	1.00	0.00	0.00	0.00	16.44	2.93	4.66	6.29
neos-3024952-loue	0.00	0.00	0.00	0.00	71	200	70	7200.04	7200.04	7200.02	7200.04	12931.00	15012.00	17198.00	14945.70	35.60	67.97	33.53	0.56	0.48	0.42	0.48
neos-3046615-murg	1.95	1.99	2.02	1.99	200	200	200	7200.08	7200.05	7200.04	7200.06	1422131.00	1459471.00	1580658.00	1485904.84	11.82	15.82	8.27	0.01	0.00	0.00	0.00
neos-3381206-awhea	0.00	0.00	0.00	0.00	200	0	132	44.48	1.61	42.22	16.25	23.00	1.00	11.00	7.32	39.81	0.00	37.91	1.93	1.61	3.84	2.33
neos-3627168-kasai	0.00	0.00	0.00	0.00	67	18	50	7200.00	1370.11	2396.28	2870.19	401233.00	47235.00	77540.00	113692.09	933.24	410.35	936.08	0.02	0.03	0.03	0.03
neos-3656078-kumeu	0.40	0.05	0.05	0.15	0	2	0	7200.13	7200.87	7200.17	7200.39	2366.00	4131.00	8671.00	4392.65	0.00	1097.70	0.00	3.04	1.74	0.83	1.73
neos-3754480-nidda	Inf	Inf	Inf	Inf	200	200	200	7200.06	7200.05	7200.04	7200.05	3860083.00	3663049.00	3668737.00	3729511.78	13.87	16.89	19.85	0.00	0.00	0.00	0.00
neos-4300652-rahue	2.40	2.30	2.72	2.47	200	69	7	7200.46	7200.16	7200.05	7200.22	423.00	1051.00	367.00	546.53	897.92	600.59	502.52	17.02	6.85	19.62	13.29
neos-4338804-snowy	0.02	0.02	0.02	0.02	0	0	0	7200.22	7200.19	7200.18	7200.20	1094222.00	1005252.00	923932.00	1005402.90	0.00	0.00	0.00	0.01	0.01	0.01	0.01
neos-4387871-tavua	0.12	0.15	0.15	0.14	0	0	0	7200.16	7200.08	7200.00	7200.08	3201.00	4561.00	4405.00	4006.50	0.00	0.00	0.00	2.25	1.58	1.63	1.81

Continued on next page

Table D.2: Branch-and-bound experiment results for VPC applied without the restart requirement. Columns r1, r2, and r3 refer to runs 1, 2, and 3 respectively. Cut Time denotes the cut generation time. Columns labeled gmean show the shifted geometric mean over the three runs.

	Final Gap (%)				#Cuts			Solve Time (s)				Node Count				Cut Time (s)			Time/Node (s)			
instance	r1	r2	r3	gmean	r1	r2	r3	r1	r2	r3	gmean	r1	r2	r3	gmean	r1	r2	r3	r1	r2	r3	gmean
neos-4647030-tutaki	0.00	0.00	0.00	0.00	200	0	0	7204.67	7200.46	7200.14	7201.75	3259.00	4387.00	4897.00	4121.57	535.83	0.00	0.00	2.21	1.64	1.47	1.76
neos-4738912-atrato	0.00	0.00	0.00	0.00	28	0	0	822.87	1025.85	1113.62	979.62	1765.00	7349.00	4365.00	3840.09	370.04	0.00	0.00	0.47	0.14	0.26	0.28
neos-4954672-berkel	0.09	0.10	0.11	0.10	69	200	200	7200.01	7200.01	7200.01	7200.01	232183.00	175239.00	160296.00	186836.33	444.88	546.42	543.35	0.03	0.04	0.04	0.04
neos-5052403-cygnet	0.01	0.01	0.01	0.01	0	2	0	7200.58	7200.67	7200.31	7200.52	794.00	741.00	598.00	705.97	0.00	1553.30	0.00	9.07	9.72	12.04	10.21
neos-5093327-huahum	0.13	0.11	0.13	0.13	0	0	0	7200.00	7200.02	7200.15	7200.05	6653.00	8098.00	5922.00	6833.16	0.00	0.00	0.00	1.08	0.89	1.22	1.06
neos-5107597-kakapo	1.13	1.08	1.17	1.12	0	0	7	7200.03	7200.02	7200.02	7200.02	282776.00	220562.00	227242.00	242002.93	0.00	0.00	203.29	0.03	0.03	0.03	0.03
neos-5188808-nattai	0.00	0.00	0.00	0.00	0	0	0	3819.04	3588.73	3072.48	3479.06	15855.00	12262.00	11991.00	13259.52	0.00	0.00	0.00	0.24	0.29	0.26	0.26
neos-5195221-niemur	0.00	0.00	0.00	0.00	200	68	0	3730.43	3445.86	3366.42	3510.82	21404.00	20048.00	25822.00	22293.87	430.65	239.69	0.00	0.17	0.17	0.13	0.16
neos-662469	0.00	0.00	0.00	0.00	74	9	67	1287.86	563.02	1280.61	975.68	1033.00	1093.00	1085.00	1070.00	981.79	291.22	980.51	1.25	0.52	1.18	0.95
neos-787933	0.00	0.00	0.00	0.00	0	0	0	2.68	2.31	2.51	2.50	1.00	1.00	1.00	1.00	0.00	0.00	0.00	2.68	2.31	2.51	2.50
neos-860300	0.00	0.00	0.00	0.00	0	0	0	19.75	18.72	16.46	18.26	1.00	1.00	1.00	1.00	0.00	0.00	0.00	19.75	18.72	16.46	18.26
neos-911970	0.00	0.00	0.00	0.00	200	200	200	148.33	136.97	163.78	149.30	133.00	171.00	249.00	178.28	116.45	116.74	119.44	1.12	0.80	0.66	0.85
neos-950242	0.00	0.00	0.00	0.00	0	0	0	503.65	474.30	511.16	496.11	134.00	27.00	43.00	53.99	0.00	0.00	0.00	3.76	17.57	11.89	9.44
neos-960392	0.00	0.00	0.00	0.00	0	0	0	63.07	90.71	94.19	81.39	1.00	1.00	1.00	1.00	0.00	0.00	0.00	63.07	90.71	94.19	81.39
neos17	0.00	0.00	0.00	0.00	200	41	200	129.51	39.37	81.48	74.75	2259.00	2330.00	1493.00	1988.15	92.75	11.92	50.80	0.06	0.02	0.05	0.04
neos5	0.00	0.00	0.00	0.00	200	200	200	2965.01	2269.07	2395.14	2525.82	1483983.00	1376391.00	1450158.00	1436135.30	10.86	11.63	10.51	0.00	0.00	0.00	0.00
neos8	0.00	0.00	0.00	0.00	0	0	0	2.01	3.05	2.84	2.60	1.00	1.00	1.00	1.00	0.00	0.00	0.00	2.01	3.05	2.84	2.60
net12	0.00	0.00	0.00	0.00	0	0	0	310.06	290.44	399.77	330.20	909.00	617.00	993.00	822.77	0.00	0.00	0.00	0.34	0.47	0.40	0.40
nexp-150-20-8-5	0.00	0.00	0.00	0.00	0	0	0	567.57	524.98	1505.98	765.69	1.00	1.00	1.00	1.00	0.00	0.00	0.00	567.57	524.98	1505.98	765.69
ns1830653	0.00	0.00	0.00	0.00	0	200	200	295.60	565.27	474.22	429.56	2916.00	2580.00	2502.00	2660.10	0.00	262.83	233.53	0.10	0.22	0.19	0.17
nu25-pr12	0.00	0.00	0.00	0.00	200	200	0	115.10	124.04	19.25	65.49	10.00	1.00	1.00	2.53	99.29	113.33	0.00	11.51	124.04	19.25	30.64
nursesched-sprint02	0.00	0.00	0.00	0.00	0	0	0	4.92	4.29	3.86	4.34	1.00	1.00	1.00	1.00	0.00	0.00	0.00	4.92	4.29	3.86	4.34
nw04	0.00	0.00	0.00	0.00	0	0	0	67.94	83.39	73.19	74.57	1.00	1.00	1.00	1.00	0.00	0.00	0.00	67.94	83.39	73.19	74.57
p200x1188c	0.00	0.00	0.00	0.00	0	0	0	12.61	8.06	9.06	9.75	1.00	1.00	1.00	1.00	0.00	0.00	0.00	12.61	8.06	9.06	9.74
pg	0.00	0.00	0.00	0.00	4	4	4	45.94	39.79	53.80	46.17	3.00	25.00	19.00	11.77	31.69	26.09	38.10	15.31	1.59	2.83	4.45
pg5_34	0.00	0.00	0.00	0.00	200	200	200	2661.41	2282.83	2740.94	2553.65	86230.00	80879.00	91559.00	86112.28	616.37	493.73	436.44	0.03	0.03	0.03	0.03
piperout-27	0.00	0.00	0.00	0.00	0	0	0	37.94	42.29	25.19	34.34	1.00	1.00	1.00	1.00	0.00	0.00	0.00	37.94	42.29	25.19	34.34
proteindesign121hz512p9	NaN	NaN	0.02	0.01	<NA>	<NA>	0	NaN	NaN	7200.54	18.31	NaN	NaN	1.00	0.26	NaN	NaN	0.00	NaN	NaN	7200.54	NaN
qap10	0.00	0.00	0.00	0.00	0	0	0	28.67	28.25	117.19	45.81	1.00	1.00	1.00	1.00	0.00	0.00	0.00	28.67	28.25	117.19	45.81
radiationm18-12-05	0.00	0.00	0.00	0.00	69	200	200	7200.01	7200.00	7200.04	7200.02	131744.00	176630.00	191385.00	164525.96	138.71	316.11	294.17	0.05	0.04	0.04	0.04
rail507	0.00	0.00	0.00	0.00	3	3	3	458.63	414.59	407.44	426.30	529.00	437.00	417.00	458.52	326.62	295.28	300.47	0.87	0.95	0.98	0.93
ran14x18-disj-8	0.00	0.02	0.00	0.01	200	200	200	5939.60	7200.00	6957.03	6675.82	222622.00	283740.00	250711.00	251123.50	530.97	375.12	372.21	0.03	0.03	0.03	0.03
rmatr100-p10	0.00	0.00	0.00	0.00	131	0	0	1126.92	391.45	398.46	560.28	623.00	693.00	735.00	682.08	953.79	0.00	0.00	1.81	0.56	0.54	0.89
rocI-4-11	0.00	0.00	0.00	0.00	0	0	0	238.96	182.72	259.47	224.61	14094.00	9328.00	13151.00	12002.21	0.00	0.00	0.00	0.02	0.02	0.02	0.02
rocII-5-11	0.77	0.77	0.77	0.77	0	0	0	7200.00	7200.00	7200.01	7200.01	174679.00	169353.00	118807.00	152040.25	0.00	0.00	0.00	0.04	0.04	0.06	0.05
rococoB10-011000	0.15	0.15	0.15	0.15	97	102	110	7200.10	7200.04	7200.02	7200.05	111097.00	115278.00	115666.00	113994.74	967.76	968.74	974.56	0.06	0.06	0.06	0.06
rococoC10-001000	0.04	0.05	0.03	0.04	80	74	82	7200.00	7200.02	7200.00	7200.01	130671.00	156508.00	96910.00	125611.15	933.70	930.69	941.56	0.06	0.05	0.07	0.06
roi2alpha3n4	0.00	0.00	0.00	0.00	13	14	11	5252.02	2486.02	2336.78	3124.82	10463.00	3226.00	3483.00	4898.95	265.01	530.81	311.48	0.50	0.77	0.67	0.64
roi5alpha10n8	NaN	0.31	0.32	0.20	<NA>	0	0	NaN	7200.51	7200.23	371.92	NaN	662.00	648.00	74.50	NaN	0.00	0.00	NaN	10.88	11.11	NaN
roll3000	0.00	0.00	0.00	0.00	0	200	0	298.72	407.31	168.72	273.88	1113.00	1267.00	930.00	1094.60	0.00	109.19	0.00	0.27	0.32	0.18	0.26
seymour	0.01	0.01	0.01	0.01	200	200	200	7200.01	7200.02	7200.02	7200.02	55321.00	51912.00	41748.00	49309.57	807.55	829.22	470.31	0.13	0.14	0.17	0.15
seymour1	0.00	0.00	0.00	0.00	185	200	200	1119.70	944.15	997.49	1017.85	735.00	2285.00	965.00	1174.74	975.51	585.41	818.73	1.52	0.41	1.03	0.94
sing326	0.00	0.00	0.00	0.00	0	0	0	7200.73	7202.48	7200.48	7201.23	882.00	31.00	1044.00	308.08	0.00	0.00	0.00	8.16	232.34	6.90	24.66
sing44	0.00	0.00	0.00	0.00	0	0	196	7200.29	7200.05	7200.64	7200.33	652.00	1274.00	1870.00	1158.22	0.00	0.00	2096.52	11.04	5.65	3.85	6.30
sorrell3	0.17	0.16	0.16	0.16	0	0	0	7199.22	7199.76	7200.12	7199.70	120.00	66.00	59.00	77.64	0.00	0.00	0.00	59.99	109.09	122.04	92.83
sp150x300d	0.00	0.00	0.00	0.00	200	200	0	9.00	10.49	87.62	20.67	96.00	138.00	12853.00	556.54	6.26	6.32	0.00	0.09	0.08	0.01	0.06
splice1k1	3.04	3.05	3.04	3.04	0	0	0	7200.03	7200.11	7200.00	7200.05	862.00	1398.00	857.00	1010.82	0.00	0.00	0.00	8.35	5.15	8.40	7.15
supportcase12	0.00	0.00	0.00	0.00	6	13	7	7200.14	7200.02	7200.12	7200.09	9827.00	11599.00	14167.00	11732.04	509.84	574.42	500.53	0.73	0.62	0.51	0.62
supportcase26	0.09	0.14	0.11	0.11	200	200	200	7200.01	7200.01	7200.01	7200.01	856734.00	554054.00	948236.00	766369.47	74.28	83.97	53.75	0.01	0.01	0.01	0.01
supportcase33	0.00	0.09	0.00	0.03	16	12	10	1232.04	7200.01	1582.03	2412.35	3790.00	108890.00	9877.00	15975.13	250.72	202.57	189.55	0.33	0.07	0.16	0.18
supportcase7	0.00	0.00	0.00	0.00	0	0	0	477.68	697.48	352.13	489.58	7.00	9.00	3.00	5.84	0.00	0.00	0.00	68.24	77.50	117.38	85.33
swath1	0.00	0.00	0.00	0.00	200	200	200	387.21	388.33	349.52	374.58	1281.00	409.00	424.00	605.76	173.39	197.38	164.04	0.30	0.95	0.82	0.67
swath3	0.00	0.00	0.00	0.00	200	200	200	1738.43	6396.04	1034.17	2257.39	25165.00	114790.00	11885.00	32501.69	183.37	334.53	237.16	0.07	0.06	0.09	0.07
timtab1	0.00	0.00	0.00	0.00	200	200	200	922.92	469.49	634.74	650.36	60217.00	31207.00	40661.00	42434.27	47.82	26.42	31.32	0.02	0.02	0.02	0.02
tr12-30	0.00	0.00	0.00	0.00	200	200	200	3024.80	3753.44	3143.68	3292.47	178480.00	186670.00	218767.00	193886.98	84.14	77.72	71.06	0.02	0.02	0.01	0.02
traininstance6	0.92	0.66	0.00	0.47	0	0	0	7200.01	7200.00	2994.23	5374.33	424759.00	669400.00	479102.00	514539.57	0.00	0.00	0.00	0.02	0.01	0.01	0.01
uccase9	0.01	0.01	0.01	0.01	85	19	0	7199.51	7200.85	7200.68	7200.35	538.00	817.00	771.00	697.21	1247.25	1041.91	0.00	13.38	8.81	9.34	10.34
uct-subprob	0.00	0.00	0.00	0.00	200	200	200	3966.15	3083.56	3022.71	3331.25	61951.00	41001.00	40331.00	46790.83	426.26	377.04	439.57	0.06	0.08	0.07	0.07
unitcal_7	0.00	0.00	0.00	0.00	0	0	6	1024.84	1197.11	1075.02	1096.65	51.00	129.00	61.00	73.84	0.00	0.00	782.46	20.09	9.28	17.62	14.92
var-smallemery-m6j6	0.01	0.01	0.01	0.01	200	200	200	7200.03	7200.00	7200.03	7200.02	139667.00	123910.00	172169.00	143897.00	151.00	158.43	112.85	0.05	0.06	0.04	0.05

Table D.3: Branch-and-bound experiment results for VPC applied only aftr at least one restart. Columns r1, r2, and r3 refer to runs 1, 2, and 3 respectively. Cut Time denotes the cut generation time. Columns labeled gmean show the shifted geometric mean over the three runs.

	Final Gap (%)				#Cuts			Solve Time (s)				Node Count				Cut Time (s)			Time/Node (s)			
instance	r1	r2	r3	gmean	r1	r2	r3	r1	r2	r3	gmean	r1	r2	r3	gmean	r1	r2	r3	r1	r2	r3	gmean
50v-10	0.01	0.01	0.01	0.01	171	188	125	7200.00	7200.03	7200.01	7200.01	127968.00	102162.00	88376.00	104932.11	922.99	918.83	928.22	0.06	0.07	0.08	0.07
CMS750_4	0.00	0.00	0.00	0.00	0	0	0	601.18	919.84	1135.12	856.26	2073.00	0.00	4913.00	215.81	0.00	0.00	0.00	0.29	inf	0.23	inf
air05	0.00	0.00	0.00	0.00	200	178	200	838.51	953.86	881.33	889.97	13.00	9.00	9.00	10.19	810.38	919.88	844.77	64.50	105.98	97.93	87.50
app1-1	0.00	0.00	0.00	0.00	4	0	4	81.80	67.96	59.57	69.19	1.00	1.00	1.00	1.00	50.14	0.00	39.66	81.80	67.96	59.57	69.19
assign1-5-8	0.05	0.05	0.06	0.05	200	0	0	7200.01	7200.01	7200.01	7200.01	1223739.00	1472771.00	1496227.00	1391897.24	36.26	0.00	0.00	0.01	0.00	0.00	0.01
atlanta-ip	0.00	0.05	0.00	0.02	0	0	0	6599.24	7200.02	6788.51	6858.05	5883.00	3036.00	3882.00	4108.24	0.00	0.00	0.00	1.12	2.37	1.75	1.70
b1c1s1	0.14	0.13	0.13	0.13	0	0	0	7200.03	7200.12	7200.02	7200.05	96536.00	86714.00	93942.00	92302.35	0.00	0.00	0.00	0.07	0.08	0.08	0.08
beasleyC3	0.00	0.00	0.00	0.00	0	0	0	25.12	24.19	22.06	23.76	1.00	1.00	1.00	1.00	0.00	0.00	0.00	25.12	24.19	22.06	23.76
binkar10_1	0.00	0.00	0.00	0.00	0	0	0	72.58	70.98	71.95	71.83	2073.00	2626.00	1911.00	2183.00	0.00	0.00	0.00	0.04	0.03	0.04	0.03
blp-ar98	0.00	0.00	0.00	0.00	18	121	93	7200.01	7200.03	7200.01	7200.02	68540.00	38447.00	65639.00	55717.27	446.77	953.75	967.60	0.11	0.19	0.11	0.13
blp-ic98	0.00	0.00	0.00	0.00	199	160	150	4692.86	4688.67	3212.30	4134.63	72529.00	44723.00	25055.00	43315.83	930.61	936.70	938.27	0.06	0.10	0.13	0.10
cod105	0.00	0.34	0.00	0.10	0	0	0	325.25	7200.02	746.65	1205.56	1411.00	45793.00	5481.00	7076.19	0.00	0.00	0.00	0.23	0.16	0.14	0.17
cost266-UUE	0.00	0.02	0.02	0.01	0	0	0	6143.46	7200.01	7200.01	6829.04	176807.00	193694.00	178064.00	182696.65	0.00	0.00	0.00	0.03	0.04	0.04	0.04
csched007	0.00	0.00	0.04	0.01	200	200	200	5717.92	5853.30	7200.01	6222.87	81749.00	85385.00	91827.00	86220.69	764.29	584.52	803.60	0.07	0.07	0.08	0.07
dano3_3	0.00	0.00	0.00	0.00	77	76	39	250.00	185.72	186.96	205.53	14.00	12.00	12.00	12.64	78.21	65.80	48.05	17.86	15.48	15.58	16.27
decomp2	0.00	0.00	0.00	0.00	0	0	0	0.11	0.11	0.10	0.11	0.00	0.00	0.00	0.00	0.00	0.00	0.00	inf	inf	inf	inf
dws008-01	0.44	0.39	0.43	0.42	200	200	200	7200.01	7200.00	7200.00	7200.00	28591.00	35254.00	27051.00	30098.19	160.10	221.30	158.03	0.25	0.20	0.27	0.24
eil33-2	0.00	0.00	0.00	0.00	200	200	200	472.73	425.12	413.86	436.51	325.00	427.00	339.00	361.00	257.27	209.96	186.69	1.45	1.00	1.22	1.22
exp-1-500-5-5	0.00	0.00	0.00	0.00	0	0	0	3.48	2.51	3.56	3.16	1.00	1.00	1.00	1.00	0.00	0.00	0.00	3.48	2.51	3.56	3.15
fast0507	0.00	0.00	0.00	0.00	3	3	2	444.00	475.87	532.43	482.74	353.00	473.00	388.00	401.63	317.02	359.19	419.76	1.26	1.01	1.37	1.21
fiball	0.00	0.00	0.00	0.00	0	0	0	4.36	3.44	4.07	3.94	1.00	1.00	1.00	1.00	0.00	0.00	0.00	4.36	3.44	4.07	3.94
gen-ip002	0.00	0.00	0.00	0.00	200	200	200	4495.22	4544.56	4214.13	4415.54	4950787.00	4877348.00	4657606.00	4826958.64	5.82	5.78	5.64	0.00	0.00	0.00	0.00
gen-ip054	0.00	0.00	0.00	0.00	200	200	200	5463.29	7200.01	3155.94	4988.59	6123732.00	8608817.00	3882106.00	5893080.20	3.95	3.21	3.84	0.00	0.00	0.00	0.00
glass-sc	0.09	0.11	0.00	0.06	200	200	200	7200.01	7200.01	6947.68	7114.90	254478.00	238765.00	280492.00	257343.85	215.60	197.87	177.14	0.03	0.03	0.02	0.03
glass4	0.00	0.00	0.00	0.00	0	0	0	2533.35	2112.96	1142.30	1828.68	522798.00	335390.00	211302.00	333372.14	0.00	0.00	0.00	0.00	0.01	0.01	0.01
gmu-35-40	0.00	0.00	0.00	0.00	200	123	200	7200.03	7200.02	7200.19	7200.08	2007067.00	1893721.00	1562171.00	1810793.03	18.61	7.98	22.22	0.00	0.00	0.00	0.00
gmu-35-50	0.00	0.00	0.00	0.00	200	200	200	7200.16	7200.02	7200.12	7200.10	799429.00	850117.00	794902.00	814436.81	12.78	21.20	13.16	0.01	0.01	0.01	0.01
graph20-20-1rand	0.00	0.48	0.00	0.14	0	0	0	5410.39	7200.01	6768.87	6412.49	120805.00	170512.00	187409.00	156871.30	0.00	0.00	0.00	0.04	0.04	0.04	0.04
graphdraw-domain	0.00	0.00	0.00	0.00	200	200	200	4885.54	3605.31	3673.42	4014.61	869723.00	750423.00	744828.00	786286.88	20.41	11.77	15.58	0.01	0.00	0.00	0.01
h80x6320d	0.00	0.00	0.00	0.00	0	0	0	221.75	229.28	270.35	239.55	9.00	5.00	1.00	3.93	0.00	0.00	0.00	24.64	45.86	270.35	67.82
hypothyroid-k1	0.00	0.00	0.00	0.00	0	0	0	34.97	32.74	36.30	34.64	0.00	0.00	0.00	0.00	0.00	0.00	0.00	inf	inf	inf	inf
ic97_potential	0.00	0.01	0.00	0.00	200	200	200	7200.00	7200.01	7200.01	7200.01	895136.00	1075976.00	956752.00	973113.87	98.22	39.14	45.07	0.01	0.01	0.01	0.01
icir97_tension	0.00	0.00	0.00	0.00	165	200	200	7200.02	7200.01	7200.01	7200.01	249209.00	253357.00	243866.00	248780.29	38.90	39.63	95.16	0.03	0.03	0.03	0.03
irp	0.00	0.00	0.00	0.00	0	0	0	16.08	14.14	15.38	15.18	1.00	1.00	1.00	1.00	0.00	0.00	0.00	16.08	14.14	15.38	15.18
istanbul-no-cutoff	0.00	0.00	0.00	0.00	0	0	0	175.98	198.69	182.93	185.63	431.00	450.00	281.00	379.16	0.00	0.00	0.00	0.41	0.44	0.65	0.50
k1mushroom	0.00	0.00	0.00	0.00	0	0	0	2839.91	2814.82	2694.96	2782.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	inf	inf	inf	inf
lectsched-5-obj	0.60	0.60	0.60	0.60	200	69	70	7200.01	7200.01	7200.03	7200.02	67548.00	68052.00	75896.00	70397.53	203.13	88.96	114.58	0.11	0.11	0.09	0.10
lotsize	0.01	0.01	0.12	0.05	0	0	0	2130.03	558.64	7200.11	2046.85	120.00	5.00	27632.00	270.72	0.00	0.00	0.00	17.75	111.73	0.26	12.86
mas74	0.00	0.00	0.00	0.00	0	200	200	4361.08	3845.32	3217.83	3778.89	4779566.00	4036598.00	3086455.00	3905002.41	0.00	8.54	10.32	0.00	0.00	0.00	0.00
mas76	0.00	0.00	0.00	0.00	200	200	200	391.87	309.70	342.73	346.48	304994.00	311406.00	291848.00	302639.07	6.87	5.17	6.23	0.00	0.00	0.00	0.00
mc11	0.00	0.00	0.00	0.00	0	0	200	308.67	226.83	387.95	300.62	1241.00	608.00	1195.00	966.14	0.00	0.00	174.27	0.25	0.37	0.32	0.31
mcsched	0.00	0.00	0.00	0.00	200	200	0	1180.48	1469.04	386.34	875.23	4685.00	4648.00	5498.00	4928.64	367.89	451.61	0.00	0.25	0.32	0.07	0.21
mik-250-20-75-4	0.00	0.00	0.00	0.00	200	200	200	233.60	269.15	360.13	282.92	8593.00	9280.00	17831.00	11244.96	24.70	32.09	21.23	0.03	0.03	0.02	0.03
milo-v12-6-r2-40-1	0.08	0.05	0.05	0.06	0	0	0	7200.07	7200.01	7200.07	7200.05	50988.00	51869.00	52396.00	51747.73	0.00	0.00	0.00	0.14	0.14	0.14	0.14
mushroom-best	0.00	0.00	0.00	0.00	0	0	0	4342.23	4758.42	6293.67	5066.34	17496.00	18864.00	21104.00	19097.62	0.00	0.00	0.00	0.25	0.25	0.30	0.27
mzzv42z	0.00	0.00	0.00	0.00	0	0	0	252.10	229.16	215.56	231.79	1.00	1.00	1.00	1.00	0.00	0.00	0.00	252.10	229.16	215.56	231.79
n3div36	0.04	0.03	0.04	0.03	137	95	157	7200.10	7200.01	7200.02	7200.04	76587.00	66299.00	69780.00	70761.61	950.38	956.77	948.76	0.09	0.11	0.10	0.10
n5-3	0.00	0.00	0.00	0.00	0	0	0	64.70	60.68	62.66	62.66	532.00	607.00	678.00	602.72	0.00	0.00	0.00	0.12	0.10	0.09	0.10
neos-1445765	0.00	0.00	0.00	0.00	0	0	0	50.56	65.50	66.18	60.30	1.00	1.00	1.00	1.00	0.00	0.00	0.00	50.56	65.50	66.18	60.30
neos-1456979	0.00	0.00	0.00	0.00	200	0	200	877.97	501.70	927.44	742.04	4481.00	1405.00	5210.00	3201.28	539.47	0.00	536.86	0.20	0.36	0.18	0.24
neos-1582420	0.00	0.00	0.00	0.00	0	0	0	8.80	3.87	6.94	6.24	1.00	1.00	1.00	1.00	0.00	0.00	0.00	8.80	3.87	6.94	6.24
neos-3024952-loue	0.00	0.00	0.00	0.00	200	0	0	7200.02	7200.04	7200.02	7200.02	15492.00	14022.00	31097.00	18903.69	332.50	0.00	0.00	0.46	0.51	0.23	0.40
neos-3046615-murg	1.94	2.08	1.89	1.97	200	200	200	7200.11	7200.11	7200.05	7200.09	1842992.00	1462822.00	1800303.00	1693116.52	21.62	19.79	17.91	0.00	0.00	0.00	0.00
neos-3381206-awhea	0.00	0.00	0.00	0.00	0	0	0	4.22	1.60	3.06	2.80	37.00	1.00	21.00	10.87	0.00	0.00	0.00	0.11	1.60	0.15	0.49
neos-3627168-kasai	0.00	0.00	0.00	0.00	66	64	200	622.21	1102.26	4060.72	1407.23	17928.00	46763.00	161439.00	51343.38	188.34	166.40	494.42	0.03	0.02	0.03	0.03
neos-3656078-kumeu	0.37	0.11	0.09	0.18	0	0	0	7199.85	7200.08	7200.03	7199.99	1912.00	6042.00	4400.00	3704.44	0.00	0.00	0.00	3.77	1.19	1.64	2.02
neos-3754480-nidda	Inf	Inf	Inf	Inf	200	200	200	7200.05	7200.04	7200.04	7200.04	4095852.00	3838877.00	4499345.00	4135860.83	35.09	80.01	48.62	0.00	0.00	0.00	0.00
neos-4300652-rahue	1.72	2.70	2.15	2.16	0	0	0	7200.46	7200.13	7200.06	7200.22	1507.00	1451.00	1521.00	1492.69	0.00	0.00	0.00	4.78	4.96	4.73	4.82
neos-4338804-snowy	0.02	0.02	0.02	0.02	0	0	0	7200.19	7200.02	7200.01	7200.07	1015233.00	1083314.00	1130119.00	1075180.60	0.00	0.00	0.00	0.01	0.01	0.01	0.01
neos-4387871-tavua	0.13	0.14	0.14	0.14	0	0	0	7200.54	7200.26	7200.50	7200.43	5733.00	4708.00	4194.00	4837.45	0.00	0.00	0.00	1.26	1.53	1.72	1.49

Continued on next page

Table D.3: Branch-and-bound experiment results for VPC applied with the requirement of at least one restart. Columns r1, r2, and r3 refer to runs 1, 2, and 3 respectively. Cut Time denotes the cut generation time. Columns labeled gmean show the shifted geometric mean over the three runs.

	Final Gap (%)				#Cuts			Solve Time (s)				Node Count				Cut Time (s)			Time/Node (s)			
instance	r1	r2	r3	gmean	r1	r2	r3	r1	r2	r3	gmean	r1	r2	r3	gmean	r1	r2	r3	r1	r2	r3	gmean
neos-4647030-tutaki	0.00	0.00	0.00	0.00	0	0	0	7200.11	7200.38	7200.01	7200.17	2188.00	3074.00	4820.00	3188.65	0.00	0.00	0.00	3.29	2.34	1.49	2.29
neos-4738912-atrato	0.00	0.00	0.00	0.00	17	0	0	1214.33	920.34	1101.85	1071.86	1724.00	4901.00	6704.00	3840.67	234.68	0.00	0.00	0.70	0.19	0.16	0.33
neos-4954672-berkel	0.10	0.08	0.11	0.10	45	200	159	7200.17	7200.01	7200.01	7200.06	133409.00	158385.00	144904.00	145208.89	599.25	647.31	910.42	0.05	0.05	0.05	0.05
neos-5052403-cygnet	0.01	0.01	NaN	0.01	0	0	<NA>	7200.28	7506.94	NaN	377.13	1580.00	1.00	NaN	13.68	0.00	0.00	NaN	4.56	7506.94	NaN	NaN
neos-5093327-huahum	0.11	0.12	0.12	0.12	0	0	0	7200.04	7200.12	7200.00	7200.05	10312.00	6672.00	7672.00	8081.71	0.00	0.00	0.00	0.70	1.08	0.94	0.90
neos-5107597-kakapo	0.87	1.21	1.24	1.10	10	5	51	7200.02	7200.01	7200.02	7200.01	217579.00	219800.00	218700.00	218691.12	352.43	43.58	509.00	0.03	0.03	0.03	0.03
neos-5188808-nattai	0.00	0.00	0.00	0.00	0	0	0	2821.47	2972.48	3358.85	3042.72	9142.00	11587.00	14866.00	11634.19	0.00	0.00	0.00	0.31	0.26	0.23	0.26
neos-5195221-niemur	0.00	0.00	0.00	0.00	0	7	6	3758.30	3570.50	3141.08	3480.17	26328.00	11976.00	9024.00	14170.31	0.00	134.46	130.03	0.14	0.30	0.35	0.26
neos-662469	0.00	0.00	0.00	0.00	68	78	76	1339.35	1211.89	1569.72	1365.82	1289.00	961.00	1639.00	1266.27	969.59	966.35	968.26	1.04	1.26	0.96	1.08
neos-787933	0.00	0.00	0.00	0.00	0	0	0	2.66	2.63	2.52	2.60	1.00	1.00	1.00	1.00	0.00	0.00	0.00	2.66	2.63	2.52	2.60
neos-860300	0.00	0.00	0.00	0.00	0	0	0	18.25	17.44	13.10	16.10	1.00	1.00	1.00	1.00	0.00	0.00	0.00	18.25	17.44	13.10	16.11
neos-911970	0.00	0.00	0.00	0.00	0	0	0	44.55	42.30	70.33	51.01	178.00	479.00	151.00	234.49	0.00	0.00	0.00	0.25	0.09	0.47	0.26
neos-950242	0.00	0.00	0.00	0.00	0	0	0	89.42	95.24	235.25	126.15	46.00	38.00	76.00	51.07	0.00	0.00	0.00	1.94	2.51	3.10	2.48
neos-960392	0.00	0.00	0.00	0.00	0	0	0	100.73	108.92	108.94	106.13	1.00	1.00	1.00	1.00	0.00	0.00	0.00	100.73	108.92	108.94	106.13
neos17	0.00	0.00	0.00	0.00	0	0	0	32.63	48.82	23.68	33.58	2093.00	4351.00	2327.00	2767.32	0.00	0.00	0.00	0.02	0.01	0.01	0.01
neos5	0.00	0.00	0.00	0.00	200	200	200	4743.18	2017.47	3292.48	3158.46	3507603.00	1059581.00	2032247.00	1962036.38	10.22	10.94	8.77	0.00	0.00	0.00	0.00
neos8	0.00	0.00	0.00	0.00	0	0	0	4.04	4.37	3.62	4.00	1.00	1.00	1.00	1.00	0.00	0.00	0.00	4.04	4.37	3.62	4.00
net12	0.00	0.00	0.00	0.00	0	0	0	207.93	314.88	260.73	257.50	764.00	1012.00	923.00	893.64	0.00	0.00	0.00	0.27	0.31	0.28	0.29
nexp-150-20-8-5	0.00	0.00	0.00	0.00	0	0	0	568.46	488.94	1078.38	669.29	1.00	1.00	1.00	1.00	0.00	0.00	0.00	568.46	488.94	1078.38	669.29
ns1830653	0.00	0.00	0.00	0.00	0	0	0	270.14	237.16	338.17	278.79	3141.00	3157.00	3851.00	3367.49	0.00	0.00	0.00	0.09	0.08	0.09	0.08
nu25-pr12	0.00	0.00	0.00	0.00	200	0	0	104.81	9.28	14.68	24.74	52.00	1.00	1.00	4.96	81.24	0.00	0.00	2.02	9.28	14.68	6.86
nursesched-sprint02	0.00	0.00	0.00	0.00	0	0	0	4.74	4.38	4.41	4.51	1.00	1.00	1.00	1.00	0.00	0.00	0.00	4.74	4.38	4.41	4.51
nw04	0.00	0.00	0.00	0.00	0	0	0	34.35	53.57	51.55	45.63	1.00	1.00	1.00	1.00	0.00	0.00	0.00	34.35	53.57	51.55	45.63
p200x1188c	0.00	0.00	0.00	0.00	0	0	0	10.30	7.97	9.10	9.08	1.00	1.00	1.00	1.00	0.00	0.00	0.00	10.30	7.97	9.10	9.08
pg	0.00	0.00	0.00	0.00	4	4	4	40.80	42.30	43.30	42.12	27.00	13.00	13.00	16.64	26.13	25.73	27.93	1.51	3.25	3.33	2.59
pg5_34	0.00	0.00	0.00	0.00	200	200	200	2199.49	2140.94	3079.23	2438.52	90278.00	75944.00	94255.00	86455.63	309.33	376.34	415.55	0.02	0.03	0.03	0.03
piperout-27	0.00	0.00	0.00	0.00	0	0	0	34.15	25.44	34.87	31.18	1.00	1.00	1.00	1.00	0.00	0.00	0.00	34.15	25.44	34.87	31.18
proteindesign121hz512p9	0.02	0.02	NaN	0.02	0	0	<NA>	7200.49	7200.49	NaN	371.92	1.00	1.00	NaN	0.59	0.00	0.00	NaN	7200.49	7200.49	NaN	NaN
qap10	0.00	0.00	0.00	0.00	0	0	0	60.57	71.69	89.15	72.89	1.00	1.00	1.00	1.00	0.00	0.00	0.00	60.57	71.69	89.15	72.89
radiationm18-12-05	0.00	0.00	0.00	0.00	93	117	149	7200.00	7200.00	7200.01	7200.01	156634.00	167801.00	204179.00	175077.59	939.49	944.75	937.21	0.05	0.04	0.04	0.04
rail507	0.00	0.00	0.00	0.00	3	3	3	398.39	396.65	370.23	388.21	519.00	367.00	365.00	411.20	296.49	299.86	282.61	0.77	1.08	1.01	0.95
ran14x18-disj-8	0.00	0.00	0.00	0.00	200	200	200	4841.62	4128.93	3231.98	4012.67	199694.00	185268.00	127908.00	167888.56	349.75	283.10	296.27	0.02	0.02	0.03	0.02
rmatr100-p10	0.00	0.00	0.00	0.00	0	0	0	152.28	191.77	201.76	180.62	757.00	815.00	697.00	754.80	0.00	0.00	0.00	0.20	0.24	0.29	0.24
rocI-4-11	0.00	0.00	0.00	0.00	0	0	0	212.82	184.80	356.37	241.14	17003.00	10701.00	20883.00	15604.46	0.00	0.00	0.00	0.01	0.02	0.02	0.02
rocII-5-11	0.76	0.77	0.77	0.77	0	0	0	7200.00	7200.01	7200.01	7200.00	158699.00	132277.00	138305.00	142658.88	0.00	0.00	0.00	0.05	0.05	0.05	0.05
rococoB10-011000	0.16	0.20	0.15	0.17	0	0	3	7200.03	7200.01	7200.05	7200.03	146585.00	140883.00	90085.00	122989.20	0.00	0.00	471.05	0.05	0.05	0.08	0.06
rococoC10-001000	0.03	0.05	0.04	0.04	0	14	84	7200.01	7200.05	3080.71	5425.59	190286.00	155477.00	37059.00	103115.42	0.00	362.54	963.19	0.04	0.05	0.08	0.06
roi2alpha3n4	0.00	0.00	0.00	0.00	0	0	0	1890.55	2012.05	2333.52	2070.52	3816.00	3239.00	4575.00	3838.29	0.00	0.00	0.00	0.50	0.62	0.51	0.54
roi5alpha10n8	NaN	0.32	0.31	0.20	<NA>	0	0	NaN	7200.12	7200.18	371.91	NaN	1550.00	1282.00	124.78	NaN	0.00	0.00	NaN	4.65	5.62	NaN
roll3000	0.00	0.00	0.00	0.00	0	0	0	257.74	112.69	186.85	175.79	1752.00	505.00	1406.00	1075.65	0.00	0.00	0.00	0.15	0.22	0.13	0.17
seymour	0.01	0.01	0.01	0.01	174	145	158	7200.00	7200.00	7200.02	7200.01	41487.00	54029.00	50411.00	48345.36	972.52	963.25	968.84	0.17	0.13	0.14	0.15
seymour1	0.00	0.00	0.00	0.00	0	0	0	108.92	108.08	130.71	115.45	877.00	677.00	885.00	806.95	0.00	0.00	0.00	0.12	0.16	0.15	0.14
sing326	0.00	0.00	0.00	0.00	0	0	0	7200.31	7200.07	7200.16	7200.18	509.00	457.00	1442.00	694.93	0.00	0.00	0.00	14.15	15.76	4.99	10.50
sing44	0.00	0.00	0.00	0.00	0	0	0	7200.32	7200.32	7200.08	7200.24	2391.00	1.00	1987.00	210.87	0.00	0.00	0.00	3.01	7200.32	3.62	50.12
sorrell3	0.17	0.17	0.18	0.17	0	0	0	7200.22	7200.39	7200.06	7200.22	214.00	260.00	339.00	266.21	0.00	0.00	0.00	33.65	27.69	21.24	27.07
sp150x300d	0.00	0.00	0.00	0.00	0	0	0	2.58	32.95	37.33	15.70	230.00	3949.00	6814.00	1837.90	0.00	0.00	0.00	0.01	0.01	0.01	0.01
splice1k1	3.02	3.02	3.03	3.02	0	0	0	7200.00	7200.00	7200.00	7200.00	1255.00	1766.00	1868.00	1605.74	0.00	0.00	0.00	5.74	4.08	3.85	4.50
supportcase12	0.00	0.00	0.00	0.00	3	6	7	7200.07	7200.11	7200.23	7200.14	12563.00	13553.00	15612.00	13852.53	582.38	499.56	478.61	0.57	0.53	0.46	0.52
supportcase26	0.13	0.09	0.08	0.10	200	200	200	7200.01	7200.00	7200.02	7200.01	801866.00	879614.00	1101499.00	919304.43	90.83	76.03	66.10	0.01	0.01	0.01	0.01
supportcase33	0.00	0.21	0.00	0.07	7	200	4	1691.05	7200.02	541.92	1876.21	9677.00	38478.00	3007.00	10384.54	72.32	886.94	36.99	0.17	0.19	0.18	0.18
supportcase7	0.00	0.00	0.00	0.00	0	0	0	238.78	229.73	231.85	233.42	7.00	7.00	5.00	6.27	0.00	0.00	0.00	34.11	32.82	46.37	37.32
swath1	0.00	0.00	0.00	0.00	200	200	200	332.74	239.67	498.75	341.40	801.00	515.00	2666.00	1032.43	142.79	114.69	145.94	0.42	0.47	0.19	0.35
swath3	0.00	0.00	0.00	0.00	200	200	200	2882.26	2553.77	2167.81	2517.56	53857.00	41105.00	32717.00	41684.09	159.27	320.71	255.26	0.05	0.06	0.07	0.06
timtab1	0.00	0.00	0.00	0.00	200	200	0	792.94	359.03	300.90	440.90	58088.00	24080.00	18922.00	29801.51	66.80	38.57	0.00	0.01	0.01	0.02	0.01
tr12-30	0.00	0.00	0.00	0.00	200	200	200	3439.89	2601.63	3075.27	3019.20	210142.00	143588.00	195327.00	180633.29	56.55	42.39	63.87	0.02	0.02	0.02	0.02
traininstance6	3.19	1.27	1.26	1.78	0	0	0	7200.02	7200.00	7200.00	7200.01	475152.00	575039.00	884191.00	622814.44	0.00	0.00	0.00	0.02	0.01	0.01	0.01
uccase9	0.01	0.01	0.01	0.01	0	0	0	7200.16	7199.55	7199.24	7199.65	817.00	1261.00	976.00	1001.85	0.00	0.00	0.00	8.81	5.71	7.38	7.20
uct-subprob	0.00	0.00	0.00	0.00	14	200	200	4111.18	3367.67	4708.03	4024.51	55540.00	29997.00	59939.00	46394.33	430.17	621.08	795.78	0.07	0.11	0.08	0.09
unitcal_7	0.00	0.00	0.00	0.00	66	0	19	1555.91	983.91	1083.51	1183.76	19.00	95.00	23.00	34.85	1290.48	0.00	757.32	81.89	10.36	47.11	34.64
var-smallemery-m6j6	0.01	0.02	0.01	0.01	200	200	200	7200.00	7200.00	7200.01	7200.00	187583.00	160447.00	133773.00	159085.72	97.09	78.87	84.59	0.04	0.04	0.05	0.05