Avoiding redundant columns by adding classical Benders cuts to column generation subproblems

Marco E. Lübbecke[1], Stephen J. Maher[2], and Jonas T. Witt[1]

[1]Lehrstuhl für Operations Research, RWTH Aachen University,

{marco.luebbecke,jonas.witt}@rwth-aachen.de

[2]College of Engineering, Mathematics and Physical Sciences, University of Exeter,

s.j.maher@exeter.ac.uk

Abstract

When solving the linear programming (LP) relaxation of a mixed-integer program (MIP) with column generation, columns might be generated that are not needed to express any integer optimal solution. Such columns are called strongly redundant and the dual bound obtained by solving the LP relaxation is potentially stronger if these columns are not generated. We introduce a sufficient condition for strong redundancy that can be checked by solving a compact LP. Using a dual solution of this compact LP we generate classical Benders cuts for the subproblem so that the generation of strongly redundant columns can be avoided. The potential of these cuts to improve the dual bound of the column generation master problem is evaluated computationally using an implementation in the branch-price-and-cut solver GCG. While their efficacy is limited on classical problems, the benefits of applying the cuts is demonstrated on structured models to which a temporal decomposition can be applied.

Key words: Benders decomposition, Dantzig-Wolfe reformulation, domain reduction, column generation

1 Introduction

Dantzig-Wolfe reformulation is a mathematical programming technique that exploits model structure within mixed integer programs (MIPs). Models that are particularly suitable for its application exhibit a constraint matrix with a bordered block diagonal form, such as

$$\min \quad \sum_{k\in K} c_k^T x^k \tag{1a}$$

$$\text{s. t.} \quad \sum_{k\in K} A_k x^k \geq b \tag{1b}$$

$$D_k x^k \geq d_k \qquad \forall k \in K \tag{1c}$$

$$x^k \in \mathbb{Z}^{n_k}_{\geq 0} \quad \forall k \in K \ . \tag{1d}$$

In the *original* model (1), we have a set K of disjoint subsystems, as given by constrains (1c), which in particular do not share any variables. The subsystems are linked by constraints (1b). For ease of exposition we initially focus on the integer program (IP) given by (1). A generalization of the results to MIPs will be presented later in this paper.

Consider subsystem $k \in K$ from problem (1) and the corresponding feasible region defined as $X^k = \{x | D_k x \geq d_k, x \in \mathbb{Z}^{n_k}_{\geq 0}\}$. All feasible solutions in X^k can be written as a binary combination of integer points $\{\mathbf{x}^p\}_{p\in P^k} \subseteq X^k$ and non-negative integer combination of integer rays $\{\mathbf{x}^r\}_{r\in R^k} \subseteq X^k$

$$x^k = \sum_{p\in P^k} \mathbf{x}^p \lambda_p + \sum_{r\in R^k} \mathbf{x}^r \lambda_r, \quad \sum_{p\in P^k} \lambda_p = 1 \quad \forall k \in K, \quad \lambda \in \mathbb{Z}^{P^k \cup R^k}_{\geq 0} \quad , \tag{2}$$

where both P^k and R^k are finite sets [18]. This type of reformulation is called the discretization approach [26].

Vanderbeck and Savelsbergh [27] refer to the $\mathbf{x}^p$ and $\mathbf{x}^r$ as generators of X^k. Each generator corresponds to a variable/column in the Dantzig-Wolfe reformulated model. Throughout this paper, we will sometimes refer to solutions $\bar{x}^k \in X^k$ as columns, although solution $\bar{x}^k$ is multiplied with A_k in order to obtain the corresponding column. Since every subsystem (1c) ignores all the other constraints, a solution in X^k (embedded in the original space) need not be feasible for (1). The generators/columns leading to such infeasible solutions are redundant in the sense that the corresponding variables take a zero value in all integer solutions to (1). Reducing the domain of X^k, and thus avoiding the generation of redundant columns is the primary focus of this paper.

A Dantzig-Wolfe reformulation of (1), using the above discretization approach is performed by substituting x^k with (2) and applying the transformations $c_j = c_k^T \mathbf{x}^j$ and $a_j = A_k \mathbf{x}^j$ for all $j \in P^k \cup R^k$.

This results in the reformulation of (1) given by the following master IP

$$\min \quad \sum_{k\in K}\left(\sum_{p\in P^k} c_p\lambda_p + \sum_{r\in R^k} c_r\lambda_r\right) \tag{3a}$$

$$\text{s.t.} \quad \sum_{k\in K}\left(\sum_{p\in P^k} a_p\lambda_p + \sum_{r\in R^k} a_r\lambda_r\right) \geq b \tag{3b}$$

$$\sum_{p\in P^k} \lambda_p = 1 \qquad \forall k \in K \tag{3c}$$

$$\lambda \in \mathbb{Z}_{\geq 0}^{P\cup R} \quad . \tag{3d}$$

The reformulated model (3) usually consists of an exponential number of so-called master variables, as given by the large cardinalities of $P = \bigcup_{k\in K} P^k$ and $R = \bigcup_{k\in K} R^k$. As a result, a delayed column generation algorithm is required to solve the LP relaxation. One starts with a restricted master LP by replacing P^k and R^k with $\bar{P}^k \subseteq P^k$ and $\bar{R}^k \subseteq R^k$, respectively, for all $k \in K$ in (3), and dropping the integrality constraints on the variables. Additional variables with negative reduced cost for the restricted master LP—either points or rays to append to $\bar{P}^k$ and $\bar{R}^k$, respectively—are identified by solving a subproblem (or pricing problem) for each subsystem k. Denoting π as the dual values associated with constraints (3b) and π_0^k as the dual value associated with constraint (3c) for subsystem k, the subproblem for $k \in K$ is given by

$$\min \quad (c_k^T - \pi^T A_k)x^k - \pi_0^k \tag{4a}$$

$$\text{s.t.} \quad D_k x^k \geq d_k \tag{4b}$$

$$x^k \in \mathbb{Z}_{\geq 0}^{n_k} \quad . \tag{4c}$$

The column generation algorithm terminates when (4) solves with a non-negative optimum for all $k \in K$. This indicates that for each $k \in K$ no feasible solution of X^k corresponds to a column in the restricted master LP with a negative reduced cost. If the solution to the restricted master LP is fractional, then branch-and-price [3] is employed to find an integer optimal solution.

A characteristic of Dantzig-Wolfe reformulation is that most master variables are zero in an integer solution. In particular, it can be observed from (2) that for each $k \in K$ exactly one λ_p, $p \in P^k$, is required to express any integer solution to the original IP, including any integer optimal solution. Given an optimal integer solution Vanderbeck and Savelsbergh [27] characterize all columns that are not required to express this solution as *redundant*:

Definition 1.1 (Redundant column [27]). *A column is redundant when the master IP admits an optimal* *integer solution that can be expressed without this column.*

It could be advantageous to avoid the generation of redundant columns while solving (3). In fact, solving (3) by column generation and avoiding all redundant columns would result in a root node dual bound that closes the entire integrality gap. Even though this is maybe too much to aim for in practice, the potential dual bound improvement from reducing the domains of the X^k motivates the research presented in this paper.

Our paper is structured as follows: The remainder of this section presents the related literature and our contribution. Section 2 uses the special case $|K| = 2$ of problem (1) to formally describe redundant columns and the methods proposed to identify and eliminate the corresponding generators from the subproblem. We present the more practically and computationally relevant alternative definition of strong redundancy. An extension of our proposals to the general case will be discussed in Section 3. The algorithm developed to replace the pricing stage of the column generation algorithm is described in Section 4. Section 5 presents our computational results to assess the effectiveness of the proposed approach. Finally, in Section 6 we conclude and point to directions for future research.

1.1 Related Work

Several concepts of redundant columns have been used in the context of column generation, and—the closely related—Lagrangian relaxation. Lübbecke and Desrosiers [17] discuss the weaker concept of columns that are redundant for the LP relaxation. This includes the idea by Sol [23] whereby a column is redundant if the corresponding constraint of the dual problem is redundant. Vanderbeck and Savelsbergh [27] are among the first to present general ideas of redundancy in the context of integer solutions. They suggest that the generation of redundant columns can be avoided with the addition of variable bounds to the subproblem that are implied from the master problem constraints. Gamrath and Lübbecke [13] extend the ideas of Vanderbeck and Savelsbergh [27] by performing domain propagation in the original problem to derive the bound changes for the subproblem.

When the reduced costs of the original variables can be computed, or at least bounded, reduced cost fixing can be used to eliminate variables from the subproblem altogether. Specifically, variables from the original problem, and thus from the subproblem, are fixed to zero if their reduced cost is greater than the current optimality gap. This results in implicitly fixing all columns for the restricted master LP that have non-zero coefficients corresponding to the fixed original variables. Beasley [4] and Ceria et al. [5] both employ this approach within a Lagrangian-based heuristic. An example of reduced cost fixing in the context of column generation is presented by Hadjar et al. [14]. Their approach is limited to subproblems formulated as shortest path problems. Extending that work, Irnich et al. [15] provide one of the most in depth investigations of reduced cost fixing for column generation. They again specialize to the case where the subproblem is a variant of a shortest path problem that can be formulated as an LP. Fahle and Sellmann [9] and Fahle et al. [8] present constraint propagation techniques from constraint programming to identify original variables that can be fixed to zero in the subproblems. The domain propagation techniques presented by Gamrath and Lübbecke [13] can in turn be viewed as a generalization of the approach of Fahle and Sellmann [9] and Fahle et al. [8].

Desrosiers and Lübbecke [7] give an example in which the objective function value of the subproblem can be bounded by the dual bound obtained from the current restricted master LP in the form of an "objective function cut." This precludes columns from being generated that would violate this bound.

In the context of the vehicle routing problem, the exact solution method proposed by Baldacci et al. [2] exploits the concept of redundant columns (even though this term is not used). The approach by Baldacci et al. [2] forms a restricted master problem by enumerating all columns with a reduced cost less than $UB - LB$, where UB and LB are valid upper and lower bounds respectively. Thus, the restricted master problem is guaranteed to contain all non-redundant columns and can be solved by branch-and-bound to find an optimal integer solution to the original problem. While an integer optimal solution can be expressed using the columns from the restricted master problem, many of the enumerated columns are redundant.

The fundamental ideas underlying the approach by Baldacci et al. [2] is formalized in the work of Rönnberg and Larsson [21]. The main results of Rönnberg and Larsson [21] provide sufficient conditions for when the current set of columns in the restricted master problem will solve the original integer program. Similar to Baldacci et al. [2], redundant columns may still exist, but it is guaranteed that at least one integer optimal solution can be expressed with the given set of columns.

Concepts related to avoiding redundant columns are employed in the integral simplex method [20, 22, 28]. Broadly, the integral simplex method aims to maintain integer feasibility at each pivot. In the context of Dantzig-Wolfe reformulation, columns, possibly redundant, are generated that are sufficient to express integer feasible solutions, not necessarily optimal. This paper differs from the proposed integral simplex algorithms [20, 22, 28] and the approaches by Baldacci et al. [2] and Rönnberg and Larsson [21] by defining necessary and sufficient conditions for identifying non-redundant columns.

1.2 Our Contribution

All the authors above report that avoiding the generation of redundant columns increased the root node dual bound and improved the overall efficiency of the column generation algorithm. However, these methods are limited, with the exception of domain propagation presented by Gamrath and Lübbecke [13], in their general applicability since they have been developed to solve specific problem types—mainly vehicle routing, knapsack, and airline crew assignment problems. In particular, a general implementation that could be tested across applications is missing. In this paper we attempt to address this lack of research by investigating how to generically reduce the subproblem domain in order to avoid the generation of redundant columns.

Our contributions are as follows: While it is computationally impractical to eliminate all redundant columns, we expect that the elimination of a subset still helps achieving a tighter root node relaxation. To this end, we (a) define the more practically relevant notion of strongly redundant columns. We then (b) devise a systematic method for the identification and elimination of such columns, that (c) is based on a novel integration of Benders decomposition and Dantzig-Wolfe reformulation. In fact, using non-trivial inequalities to reduce the subproblem domain offers more generality than all the above mentioned approaches. We (d) provide a generic implementation that is planned to be made available to the academic community. It is used in (e) a detailed computational study to evaluate the improvements in the root node dual bounds for reformulated problems. This study investigates the extent to which the application of subproblem cuts can tighten the root node relaxation for classical problems. Further, we identify problem structures that are most suitable for the application of the developed approach.

2 Strongly Redundant Columns

Definition 1.1 states that a column is redundant if there exists an optimal integer solution that can be expressed without this column. This definition of redundancy fundamentally depends on a reference optimal solution—making identifying redundant columns as difficult as solving the original problem. Thus, we would like to relax this dependency by proposing the following definition.

Definition 2.1 (Strongly redundant column)**.** *A column is strongly redundant if all optimal integer solutions to the master IP can be expressed without this column, i.e., the corresponding master variable is set to zero in all optimal integer solutions.*

Clearly, strongly redundant columns are also redundant (see Definition 1.1). While previous approaches were developed to eliminate redundant columns, they are in fact eliminating only strongly redundant columns. An example is the fixing of original variables to zero if their reduced cost is strictly greater than the current optimality gap. Master LP solutions expressed using columns with non-zero coefficient corresponding to such original variables would have an objective function value greater than the optimum, which cannot be and thus such columns do not appear in any optimal solution. Although Definition 1.1 is more general, this suggests that Definition 2.1 is more practically relevant.

2.1 Identifying Strongly Redundant Columns

Consider the special case of (1) with $|K| = 2$:

$$\begin{aligned}
\min \quad & c_1^T x^1 + c_2^T x^2 && \text{(5a)}\\
\text{s. t.} \quad & A_1 x^1 + A_2 x^2 \geq b && \text{(5b)}\\
& D_1 x^1 \geq d_1 && \text{(5c)}\\
& D_2 x^2 \geq d_2 && \text{(5d)}\\
& x^1 \in \mathbb{Z}_{\geq 0}^{n_1} && \text{(5e)}\\
& x^2 \in \mathbb{Z}_{\geq 0}^{n_2} \ . && \text{(5f)}
\end{aligned}$$

The special case given by Problem (5) is used throughout this section to simplify the discussion related to the identification redundant columns and the methods proposed to avoid their generation. We extend our methods to the general case in Section 3.

Assume that a Dantzig-Wolfe reformulation is applied to the original problem (5) using the discretization approach, resulting in the master IP (3) (with $|K| = 2$) and a subproblem in form of (4). For the following results, recall that $X^k = \{x^k \in \mathbb{Z}_{\geq 0}^{n_k}, D_k x^k \geq d_k\}$ defines the set of feasible solutions X^k to Problem (4) for subsystem k.

Strongly redundant columns for the master IP (3) can be characterized as follows:

Lemma 2.1. *Given a column $\bar{x}^1 \in X^1$ and the optimum z^* of the original problem (5). Column $\bar{x}^1$ is strongly redundant if and only if there does not exist any $\bar{x}^2 \in X^2$ with $A_1\bar{x}^1 + A_2\bar{x}^2 \geq b$ and $c_1^T\bar{x}^1 + c_2^T\bar{x}^2 \leq z^*$.*

Proof. By Definition 2.1, column $\bar{x}^1$ is strongly redundant if the corresponding master variable is set to zero in all optimal solutions to the master IP (3) (with $|K| = 2$). Since the optimum z^* of the original problem (5) is equal to the optimum of the master IP (3) (with $|K| = 2$), this holds if and only if there does not exist any $\bar{x}^2 \in X^2$ with $A_1\bar{x}^1 + A_2\bar{x}^2 \geq b$ and $c_1^T\bar{x}^1 + c_2^T\bar{x}^2 \leq z^*$. $\square$

Symmetrically, we can check whether $\bar{x}^2 \in X^2$ is redundant. For ease of exposition we will only consider the case used in Lemma 2.1, i.e., the redundancy of columns in X^1.

We can identify redundant columns using the following *verification IP* that is formulated to be infeasible if and only if the column $\bar{x}^1 \in X^1$ is strongly redundant:

$$\begin{aligned}
\min \quad & c_2^T x^2 && \text{(6a)}\\
\text{s. t.} \quad & A_2 x^2 \geq b - A_1 \bar{x}^1 && \text{(6b)}\\
& D_2 x^2 \geq d_2 && \text{(6c)}\\
& c_2^T x^2 \leq z^* - c_1^T \bar{x}^1 && \text{(6d)}\\
& x^2 \in \mathbb{Z}^{n_2}_{\geq 0} \ . && \text{(6e)}
\end{aligned}$$

Note that verification IP (6) could be formulated as a feasibility problem since constraint (6d) makes the objective function unnecessary. However, the objective function will be important for results presented later in this section.

We state the desired properties of the verification IP as a lemma:

Lemma 2.2. *Given a column $\bar{x}^1 \in X^1$ and the optimum z^* of the original problem* (5). *Column $\bar{x}^1$ is strongly redundant if and only if Problem* (6) *is infeasible.*

Proof. Follows directly from Lemma 2.1. □

Checking strong redundancy of a column with Lemma 2.2 is called the *redundancy check*. Even though $\bar{x}^1$ is fixed, proving infeasibility or finding a feasible solution to (6) can still be as difficult as solving the original problem (5).

We can reduce the effort required to perform a redundancy check by relaxing the verification IP, which corresponds to relaxing the redundancy check. We state this as a lemma:

Lemma 2.3. *Given a column $\bar{x}^1 \in X^1$ and a relaxation of Problem* (6). *If the given relaxation of Problem* (6) *is infeasible, column $\bar{x}^1$ is strongly redundant.*

Proof. Clearly, if the given relaxation of Problem (6) is infeasible, Problem (6) itself is infeasible and hence, by Lemma 2.2, column $\bar{x}^1$ is strongly redundant. □

Note that Lemma 2.3 states only a sufficient condition for strongly redundant columns, while the condition in Lemma 2.2 is also necessary. Hence, with Lemma 2.3 we can only identify a subset of strongly redundant columns. We expect, however, that avoiding even only the generation of a subset of strongly redundant columns suffices to improve the dual bound obtained from the master LP.

Any relaxation of Problem (6) can be used in Lemma 2.3 to obtain a sufficient condition for strong redundancy. The following relaxation proves particularly useful. We first relax integrality on the x^2 variables. Next, to eliminate the need for finding the optimum z^* of the original problem (5) we replace z^* with the best known primal bound $\bar{z}^{UB}$ for the original problem (5). This primal bound is finite e.g., when primal heuristics were successful, but could initially be infinite. The resulting relaxation of verification IP (6), called *verification LP*, is given by

$$\begin{aligned} \min \quad & c_2^T x^2 && \text{(7a)}\\ \text{s.t.} \quad & A_2 x^2 \geq b - A_1 \bar{x}^1 && \text{(7b)}\\ & D_2 x^2 \geq d_2 && \text{(7c)}\\ & c_2^T x \leq \bar{z}^{UB} - c_1^T \bar{x} && \text{(7d)}\\ & x^2 \in \mathbb{R}^{n_2}_{\geq 0} \ . && \text{(7e)} \end{aligned}$$

Using verification LP (7) and Lemma 2.3, the following sufficient condition for identifying strongly redundant columns can be specified.

Theorem 2.1. *Given a column $\bar{x}^1 \in X^1$ and an upper bound $\bar{z}^{UB}$ on the optimum z^* of the original problem* (5)*. If Problem* (7) *is infeasible, column $\bar{x}^1$ is strongly redundant.*

Proof. First, by removing the integrality constraints on x^2 the resulting problem is the LP relaxation of verification IP (6). By Lemma 2.3, column $\bar{x}^1$ is strongly redundant if the LP relaxation of verification IP (6) is infeasible. Second, since z^* is replaced by $\bar{z}^{UB}$ in (6d) it is sufficient to prove that $c_1^T \bar{x}^1 + c_2^T \bar{x}^2 > \bar{z}^{UB}$ implies $c_1^T \bar{x}^1 + c_2^T \bar{x}^2 > z^*$; this holds because $\bar{z}^{UB} \geq z^*$. $\square$

2.2 Avoiding the Generation of Strongly Redundant Columns

To simplify the description of methods for avoiding the generation of strongly redundant columns, we will use the following alternative notation for verification LP (7):

$$\begin{aligned} \min \quad & c_2^T x^2 && \text{(8a)}\\ \text{s.t.} \quad & \tilde{B} x^2 \geq \tilde{b} - \tilde{A} \bar{x}^1 && \text{(8b)}\\ & c_2^T x^2 \leq \bar{z}^{UB} - c_1^T \bar{x}^1 && \text{(8c)}\\ & x^2 \in \mathbb{R}^{n_2}_{\geq 0} \ , && \text{(8d)} \end{aligned}$$

where

$$\tilde{A} := \begin{pmatrix} A_1 \\ 0 \end{pmatrix}, \tilde{B} := \begin{pmatrix} A_2 \\ D_2 \end{pmatrix}, \tilde{b} := \begin{pmatrix} b \\ d_2 \end{pmatrix} \ .$$

The dual of verification LP (8) will be helpful for the following methods and results. For completeness, the dual problem is given by

$$\max \quad w^T\left(\tilde{b} - \tilde{A}\bar{x}^1\right) + w^c\left(c_1^T\bar{x}^1 - \bar{z}^{UB}\right) \tag{9a}$$

$$\text{s.t.} \quad w^T\tilde{B} - w^c c_2^T \leq c_2^T \tag{9b}$$

$$w \in \mathbb{R}^m_{\geq 0} \tag{9c}$$

$$w^c \in \mathbb{R}_{\geq 0} \ . \tag{9d}$$

Let T^c denote the set of *dual rays* of verification LP (8), i.e., the set of solutions to the homogeneous version of the dual problem (9)

$$T^c := \{(\tilde{w}, \tilde{w}^c) \in \mathbb{R}^{m+1}_{\geq 0} : \tilde{w}^T\tilde{B} - \tilde{w}^c c_2^T \leq 0\} \ . \tag{10}$$

Farkas' Lemma [10] connects the feasibility of verification LP (8) (and hence, the redundancy of $\bar{x}$) with the existence of certain dual rays. It states that Problem (8) is infeasible if and only if there exists a dual ray $(\tilde{w}, \tilde{w}^c) \in T^c$ of verification LP (8) with positive dual objective function value, i.e., with

$$\tilde{w}^T(\tilde{b} - \tilde{A}\bar{x}^1) + \tilde{w}^c(c_1^T\bar{x}^1 - \bar{z}^{UB}) > 0 \ . \tag{11}$$

Hence, solution $\bar{x}^1$ inducing an infeasible instance of verification LP (8) can be eliminated from the corresponding subproblem (4) by adding

$$\tilde{w}^T(\tilde{b} - \tilde{A}x^1) + \tilde{w}^c(c_1^T x^1 - \bar{z}^{UB}) \leq 0 \ , \tag{12}$$

which corresponds to a classical Benders feasibility cut. Inequality (12) is called a *subproblem cut.* It can only be violated by solutions to the subproblem corresponding to strongly redundant columns:

Theorem 2.2. *Let $(\tilde{w}, \tilde{w}^c) \in T^c$ be a dual ray of Problem (8). Inequality (12) is valid for all solutions $\bar{x}^1 \in X^1$ corresponding to columns that are not strongly redundant.*

Proof. Let $\bar{x}^1 \in X^1$ be a solution to the subproblem corresponding to a column that is not strongly redundant. Hence, $\bar{x}^1$ induces a feasible instance of (8) and the corresponding dual problem (9) is bounded, i.e., the objective function value of all dual rays in T^c are non-positive. In particular, this holds for the given dual ray $(\tilde{w}, \tilde{w}^c)$, i.e., Inequality (12) holds for $\bar{x}^1$. □

We will distinguish between two types of dual rays $(\tilde{w}, \tilde{w}^c) \in T^c$: Dual rays with $\tilde{w}^c = 0$, corresponding to instances of verification LP (8) that remain infeasible when Inequality (8c) is removed, and dual rays with $\tilde{w}^c > 0$, only occurring in instances of verification LP (8) that are feasible when Inequality (8c) is removed. When generating Inequality (12) using a dual ray of the first type, the variable corresponding to $\bar{x}^1$ is set to zero in all *feasible* solutions to the master IP. In this case the resulting Inequality (12) is called a *feasibility (subproblem) cut.* Dual rays of the second type can also occur if the variable corresponding to $\bar{x}^1$ is not set to zero in all feasible solutions to the master IP. Nevertheless, taking the objective function into account by considering Inequality (8c), we can conclude that it is set to zero in all *optimal* solutions to the master IP. In this case, Inequality (12) is called an *optimality (subproblem) cut* and can be written as

$$c_1^T x^1 + \frac{\tilde{w}}{\tilde{w}^c}(\tilde{b} - \tilde{A}x^1) \leq \bar{z}^{UB} \quad . \tag{13}$$

The reader may note that in our quest to avoid the generation of (strongly) redundant columns, as a by-product, we additionally avoid certain columns from one subproblem that are *incompatible* with solutions of another, when using feasibility subproblem cuts.

2.3 Practical Considerations for the Generation of Subproblem Cuts

The theory presented in Section 2.2 shows the relationship between verification LP (8) and the generation of subproblem cuts to avoid the generation of redundant columns. While the results from Section 2.2 can be directly applied to avoid the generation of redundant columns, there are practical aspects of Dantzig-Wolfe and Benders decomposition that when considered can simplify the implementation effort and improve the effectiveness of the developed algorithms.

The most important practical considerations are related to the formulation and implementation of the verification LP and the efficacy of the subproblem cuts. First, it can be observed that verification LP (8) comprises the same variables and constraints as the LP relaxation of the original IP with the addition of Inequality (8c). Thus, relaxing Inequality (8c) enables the use of the LP relaxation of the original IP—with variables fixed to values corresponding to a subproblem solution—to identify redundant columns. This observation greatly simplifies the implementation of the results presented in Section 2.2. Second, our experience with Benders decomposition has indicated that optimality cuts tend to be stronger than feasibility cuts. This is due to the fact that infeasible LPs have an infinite number of dual extreme rays and there is no clear method to identify the best for generating feasibility cuts. Relaxing Inequality (8c) will enable the generation of optimality cuts in the event that infeasbility is caused by the violation of the best known upper bound—leading to more efficacious cuts. The remainder of this section will show that relaxing Inequality (8c) to address these considerations provides a practical verification LP that is as effective as verification LP (8).

Remember that feasibility subproblem cuts are generated if verification LP (8) remains infeasible when Inequality (8c) is removed. Hence, it suffices to consider the following problem:

$$\begin{aligned} \min \quad & c_2^T x^2 && (14a)\\ \text{s.t.} \quad & \tilde{B}x^2 \geq \tilde{b} - \tilde{A}\bar{x}^1 && (14b)\\ & x^2 \in \mathbb{R}^{n_2}_{\geq 0} \ , && (14c) \end{aligned}$$

and the corresponding dual of (14):

$$\begin{aligned} \max \quad & w^T \left(\tilde{b} - \tilde{A}\bar{x}^1\right) && (15a)\\ \text{s.t.} \quad & w^T \tilde{B} \leq c_2^T && (15b)\\ & w \in \mathbb{R}^{m}_{\geq 0} \ . && (15c) \end{aligned}$$

Analogously to Problem (8), we define the set of dual rays of Problem (14), i.e., the set of solutions to the homogeneous version of the dual problem (15) as

$$T := \{\tilde{w} \in \mathbb{R}^m_{\geq 0} : \tilde{w}^T \tilde{B} \leq 0\} \ . \tag{16}$$

Note that column $\bar{x}^1 \in X^1$ is strongly redundant if Problem (14) is either (i) infeasible or (ii) feasible but the corresponding objective function value is greater than the best known upper bound $\bar{z}^{UB}$.

On the one hand, if Problem (14) is infeasible, feasibility cuts can be generated from its dual rays. Let $\tilde{w} \in T$ be a dual ray of an infeasible instance of (14); the corresponding cut takes the form of a feasibility cut (12) with $\tilde{w}^c = 0$, i.e.,

$$\tilde{w}^T(\tilde{b} - \tilde{A}\bar{x}^1) \leq 0 \ . \tag{17}$$

Theorem 2.3. *Let $\tilde{w} \in T$ be a dual ray of (14). Inequality (17) is valid for all solutions $\bar{x}^1 \in X^1$ corresponding to columns that are not strongly redundant.*

Proof. Let $\bar{x}^1 \in X^1$ be a solution to the subproblem corresponding to a column that is not strongly redundant. Hence, $\bar{x}^1$ induces a feasible instance of Problem (8). Furthermore, Problem (14) is feasible and the corresponding dual problem (15) is bounded, i.e., the objective function value of all dual rays in T is non-positive. In particular, this holds for the given dual ray $\tilde{w}$, i.e., Inequality (17) holds for $\bar{x}^1$. □

On the other hand, if Problem (14) is feasible, a dual solution to (14) can be used to generate a cut similar to (13). Let S be the set of dual solutions to (14) (the set of feasible solutions to the dual (15)), i.e.,

$$S := \{\tilde{w} \in \mathbb{R}^m_{\geq 0} : \tilde{w}^T \tilde{B} \leq c_2^T\} \ . \tag{18}$$

Assume that $\bar{x}^1$ induces a feasible instance of (14). By applying weak duality, each dual solution $\tilde{w}$

to (14) induces a lower bound on the optimum of (14), i.e.,

$$\min\left\{c_2^T x^2 : \tilde{B}x^2 \geq \tilde{b} - \tilde{A}\bar{x}^1, x^2 \in \mathbb{R}^{n_2}_{\geq 0}\right\} \geq \tilde{w}(\tilde{b} - \tilde{A}\bar{x}^1) \ . \tag{19}$$

Furthermore, if $\bar{x}^1$ is not strongly redundant, then it is possible to derive a lower bound on the optimum z^* of the original problem (5) from Equation (19):

$$\begin{aligned} z^* &\geq c_1^T\bar{x}^1 + \min\left\{c_2^T x^2 : \tilde{B}x^2 \geq \tilde{b} - \tilde{A}\bar{x}^1, x^2 \in \mathbb{Z}^{n_2}\right\} && \text{(20a)} \\ &\geq c_1^T\bar{x}^1 + \min\left\{c_2^T x^2 : \tilde{B}x^2 \geq \tilde{b} - \tilde{A}\bar{x}^1, x^2 \in \mathbb{R}^{n_2}\right\} && \text{(20b)} \\ &\geq c_1^T\bar{x}^1 + \tilde{w}(\tilde{b} - \tilde{A}\bar{x}^1) \ . && \text{(20c)} \end{aligned}$$

Given an upper bound $\bar{z}^{UB}$ on the optimum z^* of (5) and a dual solution $\tilde{w} \in S$ of Problem (14), we can derive the following valid inequality to cut-off strongly redundant columns:

$$c_1^T x^1 + \tilde{w}(\tilde{b} - \tilde{A}x^1) \leq \bar{z}^{UB} \ . \tag{21}$$

Such inequalities, because of their form and role being similar to optimality cuts (13), are also called *optimality (subproblem) cuts.*

Theorem 2.4. *Let $\tilde{w} \in S$ be a dual solution of* (14)*. Inequality* (21) *is valid for all solutions $\bar{x}^1 \in X^1$ corresponding to columns that are not strongly redundant.*

Proof. Let $\bar{x}^1 \in X^1$ be a solution to the subproblem corresponding to a column that is not strongly redundant. Hence, $\bar{x}^1$ induces a feasible instance of (8). Furthermore, Problem (14) is feasible and $\bar{x}^1$ is part of an optimal integer solution to the original problem (5):

$$z^* = c_1^T\bar{x}^1 + \min\left\{c_2^T x^2 : \tilde{B}x^2 \geq \tilde{b} - \tilde{A}\bar{x}^1, x^2 \in \mathbb{Z}^{n_2}\right\} \ .$$

With (20) and $\bar{z}^{UB} \geq z^*$ this implies:

$$\begin{aligned} \bar{z}^{UB} &\geq z^* \\ &= c_1^T\bar{x}^1 + \min\left\{c_2^T x^2 : \tilde{B}x^2 \geq \tilde{b} - \tilde{A}\bar{x}^1, x^2 \in \mathbb{Z}^{n_2}\right\} \\ &\geq c_1^T\bar{x}^1 + \tilde{w}(\tilde{b} - \tilde{A}\bar{x}^1) \ . \end{aligned}$$

Thus, Inequality (21) holds for $\bar{x}^1$. □

Instead of using verification LP (8), we can use the more practical Problem (14) to perform the relaxed redundancy check for each column. Thus, we call Problem (14) the practical verification LP. The only difference between the two is the omission of the upper bound constraint (8c). Subproblem cuts (12) are generated using dual rays of infeasible instances of verification LP (8); subproblem cuts (17) and (21) are generated using dual rays of infeasible instances and dual solutions to feasible instances of (14), respectively. In the following we prove the equivalence of subproblem cuts generated using Problems (8) and (14). We start with the feasibility cuts.

Theorem 2.5. *Let $\bar{x}^1 \in X^1$. There exists a dual ray $(\tilde{u}, \tilde{u}^c) \in T^c$ of Problem (8) with $\tilde{u}^c = 0$ satisfying*

$$\tilde{u}(\tilde{b} - \tilde{A}\bar{x}^1) + \tilde{u}^c(c_1^T\bar{x}^1 - \bar{z}^{UB}) > 0 \tag{23}$$

if and only if there exists a dual ray $\tilde{w} \in T$ of Problem (14) satisfying

$$\tilde{w}^T(\tilde{b} - \tilde{A}\bar{x}^1) > 0 \ .$$

Proof. When fixing variable w^c to zero in the dual problem (9), the resulting problem is equivalent to the dual problem (15). Hence, the set of dual rays T of Problem (15) is identical to the set $\{\tilde{u} : (\tilde{u}, \tilde{u}^c) \in T^c, \tilde{u}^c = 0\}$. This proves the theorem. □

We next show the equivalence of optimality cuts derived from Problems (8) and (14).

Theorem 2.6. *Let $\bar{x}^1 \in X^1$. There exists a dual ray $(\tilde{u}, \tilde{u}^c) \in T^c$ of Problem (8) with $\tilde{u}^c > 0$ satisfying*

$$\tilde{u}(\tilde{b} - \tilde{A}\bar{x}^1) + \tilde{u}^c(c_1^T\bar{x}^1 - \bar{z}^{UB}) > 0 \tag{24}$$

if and only if there exists a dual solution $\tilde{w} \in S$ of Problem (14) satisfying

$$c_1^T x^1 + \tilde{w}(\tilde{b} - \tilde{A}x^1) > \bar{z}^{UB} \ . \tag{25}$$

Proof. Let $(\tilde{u}, \tilde{u}^c) \in T^c$ be a dual ray of Problem (8) with $\tilde{u}^c > 0$ satisfying (24). We define $\tilde{w} := \frac{\tilde{u}}{\tilde{u}^c}$. Obviously, $\tilde{w} \geq 0$ holds. Furthermore, a ray of problem (9), given by $\tilde{u}^T\tilde{B} - \tilde{u}^c c_2^T \leq 0$, implies

$$\tilde{w}^T\tilde{B} = \frac{\tilde{u}^T\tilde{B}}{\tilde{u}^c} \leq \frac{\tilde{u}^c c_2^T}{\tilde{u}^c} = c_2^T \ .$$

Hence, solution $\tilde{w}$ is a dual solution of Problem (14), i.e., $\tilde{w} \in T$. By construction, Inequality (25) holds.

Let $\tilde{w} \in S$ be a dual solution of Problem (14) satisfying (25). We define $\tilde{u}^c := 1$ and $\tilde{u} := \tilde{w}$. Obviously, $\tilde{u}^c \geq 0$ and $\tilde{u} \geq 0$ hold. Furthermore, Equation (15b) implies

$$\tilde{u}^T\tilde{B} - \tilde{u}^c c_2^T = \tilde{w}^T\tilde{B} - c_2^T \leq 0 \ .$$

Hence, $(\tilde{u}, \tilde{u}^c)$ is a dual ray of Problem (8). By construction, Inequality (23) holds. □

Although the sets of optimality cuts generated by solving the verification LP (8) and practical verification LP (14) are equally strong (c.f. Theorem 2.6), there is a difference between these optimality cuts: Optimality cuts generated by solving practical verification LP (14) maximize the violation of the considered solution $\bar{x}^1$, while optimality cuts generated by solving verification LP (8) in general do not. Note that the violation of Inequality (13) is increased if the corresponding objective function value of Problem (15), the dual of practical verification LP (14), is increased; this objective function value is maximized if practical verification LP (14) is minimized.

By maximizing the violation at the considered solution, we expect to obtain more efficacious optimality cuts, which are violated by potentially more solutions $x^1 \in X^1$. This is why we will focus on the use of the practical verification LP to generate subproblem cuts for the remainder of this paper.

2.4 The Relationship Between Reduced Cost Fixing and Subproblem Cuts

As explained in Section 1.1, a concept related to avoiding the generation of redundant columns is reduced cost fixing. The relationship between subproblem cuts and reduced cost fixing is connected to how these two concepts exploit LP duality. Using the results presented in Section 2.3 and Theorem 2.3, we will show that the arguments used to perform reduced cost fixing can be derived from the generation of subproblem cuts.

In this section we use an original problem given in the standard form:

$$\min \quad c^T x \tag{26a}$$

$$\text{s.t.} \quad Ax \geq b \tag{26b}$$

$$x \in \mathbb{Z}^n_{\geq 0} \tag{26c}$$

While the standard form is used in this section to explain the relationship between subproblem cuts and reduced cost fixing, the presented results can be simply extended to original problems in the form of problem (1). Let $x := (x_1, x_2, x_3, \dots, x_{n_1})$ and define $\hat{x}_i$ as the vector x without element i:

$$\hat{x}_i := x \backslash x_i := (x_1, \dots x_{i-1}, x_{i+1}, \dots, x_{n_1}) \ . \tag{27}$$

Similarly, we define $\hat{A}_i$ as the constraint matrix A without column i. Column i from constraint matrix A is denoted by $A(i)$. Finally, the cost vector $\hat{c}_i^T$ is equivalent to c^T without entry i and $c(i)$ is element i from c. We also define w as the vector of dual variables corresponding to the rows of (26b).

Consider a variable x_i that could be fixed to zero by reduced cost fixing. This implies that $x_i = 0$ in an optimal solution to (26). At the same time, the dual solution w given by solving the LP relaxation of (26) after setting $x_i = 0$ satisfies the inequality

$$c(i) - w^T A(i) > z^{UB} - w^T b, \tag{28}$$

where z^{UB} is some known upper bound on (26). Note that the left-hand side of Inequality (28) corresponds to the reduced cost of variable x_i and the right-hand side is the optimality gap (since $w^T b$ is a lower bound on (26)). Since the reduced cost of variable x_i exceeds the optimality gap, the variable takes non-zero values only in sub-optimal solutions; hence, it can be fixed to zero.

An inequality based on reduced cost arguments that fixes x_i to zero can be derived from the procedure used for generating subproblem cuts. This connection between subproblem cuts and the derivation of variable bound constraints is the source of the relationship between reduced cost fixing and avoiding the generation of redundant columns.

The following theorem formalizes this relationship.

Theorem 2.7. *Given a variable x_i, which can be fixed to zero by reduced cost fixing, then a subproblem cut can be generated to eliminate all solutions where $x_i \geq 1$ by solving verification LP with x_i fixed to 1.*

Proof. Since x_i can be fixed to zero by reduced cost fixing, there exists a dual solution w of the LP relaxation of (26) satisfying Inequality (28). Consider the verification LP given by

$$\min \quad \hat{c}_i^T \hat{x}_i, \tag{29a}$$

$$\text{s.t.} \quad \hat{A}_i \hat{x}_i \geq b - A(i)\bar{x}_i, \tag{29b}$$

$$-\hat{c}_i^T \hat{x}_i \geq c(i)\bar{x}_i - z^{UB}, \tag{29c}$$

$$\hat{x}_i \in \mathbb{Z}^n_{\geq 0}, \tag{29d}$$

where $\bar{x}_i = 1$. We can use w as dual values corresponding to Constraints (29b) and set the dual value w_c corresponding to Constraint (29c) to 1. Inequality (28) implies that these dual values satisfy:

$$c(i) - w^T A(i) + w^T b - z^{UB} > 0 \ . \tag{30}$$

Thus, the dual values $(w, 1)$ are a Farkas ray proving the infeasibility of (29). Hence, we can derive the following subproblem cut

$$c(i)x_i - w^T A(i)x_i - z^{UB} + w^T b \leq 0 \ , \tag{31}$$

which is equivalent to

$$x_i \leq \frac{z^{UB} - w^T b}{c(i) - w^T A(i)} \ . \tag{32}$$

Since the right-hand side of (32) is smaller than one, subproblem cut (31) eliminates all solutions $x_i \geq 1$

from (26). □

Theorem 2.7 shows that the generation of subproblem cuts derives a variable bound constraint that has the same effect on the domain of x_i as when performing reduced cost fixing. In fact, the addition of Inequality (31), which is equivalent to $x_i \leq \epsilon$ where $0 \leq \epsilon < 1$, to the column generation subproblem will result in fixing $x_i = 0$ through the application of domain propagation.

While Theorem 2.7 shows the relationship between reduced cost fixing and avoiding redundant columns, the two approaches are not computationally equivalent. The generation of Inequality (31) requires determining whether fixing variable x_i to 1 induces an infeasible instance of the verification LP. Further, if setting $x_i = 1$ induces an infeasible instance of the verification LP, then it would be much more convenient to include the constraint $x_i = 0$ to the subproblem to eliminate this variable from all subsequent columns, instead of applying Inequality (31) and relying on domain propagation to derive the same bound.

The results of this section show that the concept of reduced cost fixing need not be restricted to single master problem variables, but combinations of variables that form feasible columns. The subproblem cuts developed in this paper are a generalization of reduced cost fixing, whereby columns are eliminated from the subproblem through the addition of inequalities. While in this paper we focus on generating subproblem cuts for eliminating full solutions from the column generation subproblem, it is possible to apply this method in a more general way. One such approach could be to fix variables in the verification LP that correspond to partial solutions of the column generation subproblem and then generate corresponding subproblem cuts. Such cuts may be stronger than those presented in this paper since the same partial subproblem solutions may arise in many full feasible solutions. Investigating the generation of subproblem cuts from partial subproblem solutions is left for future work.

3 Generalization of the Proposed Methods

In this section, we consider generalizations of the proposed methods to problems with multiple subproblems in Section 3.1 and problems with continuous variables in Section 3.2.

3.1 Multiple Subproblems

The methods developed for avoiding the generation of redundant columns in Sections 2.1–2.2 focus on the special case of the original problem (1) with $|K| = 2$. From our presentation it should be clear how these generalize to $|K| > 2$. For completeness, we state the identification of redundant columns and the methods for avoiding their generation in the general case. We also present the general formulation of the verification LP used for the redundancy check and the form of the subproblem cuts. Further, we restate Theorems 2.1, 2.3, and 2.4 with respect to the general formulation of Problem (1). We focus on the generalization of subproblem cuts obtained by solving practical verification LP (14). The generalization of subproblem cuts obtained by solving verification LP (8) can be accomplished analogously.

Consider the k-th subproblem (4) for some $k \in K$ and the column corresponding to some solution $\bar{x}^k \in X^k$. The redundancy of $\bar{x}^k$ can be evaluated by checking the conditions of Theorem 2.1. This is achieved by solving the following LP, which is a generalization of practical verification LP (14):

$$\begin{aligned} z''(\bar{x}^k) = \min \quad & \sum_{k' \in K \setminus \{k\}} c_{k'}^T x^{k'} && \text{(33a)} \\ \text{s.t.} \quad & \sum_{k' \in K \setminus \{k\}} A_{k'} x^{k'} \geq b - A_k \bar{x}^k && \text{(33b)} \\ & D_{k'} x^{k'} \geq d_{k'} \qquad \forall k' \in K \setminus \{k\} && \text{(33c)} \\ & x^{k'} \in \mathbb{R}_{\geq 0}^{n_{k'}} \qquad \forall k' \in K \setminus \{k\} \ . && \text{(33d)} \end{aligned}$$

We will also call Problem (33) practical verification LP. Note that it includes all constraints related to subproblems $k' \in K \setminus \{k\}$. Constraints (33b) correspond to the linking constraints (1b); the variables x^k, however, are fixed to the values of $\bar{x}^k$.

The dual of practical verification LP (33), is given by

$$\begin{aligned} \max \quad & (w^0)^T (b - A_k \bar{x}^k) + \sum_{k' \in K \setminus \{k\}} (w^{k'})^T d_{k'} && \text{(34a)} \\ \text{s.t.} \quad & (w^0)^T A_{k'} + (w^{k'})^T D_{k'} \leq c_{k'}^T \qquad \forall k' \in K \setminus \{k\} && \text{(34b)} \\ & w^0 \in \mathbb{R}_{\geq 0}^{m} && \text{(34c)} \\ & w^{k'} \in \mathbb{R}_{\geq 0}^{m_k} \qquad \forall k' \in K \setminus \{k\} \ . && \text{(34d)} \end{aligned}$$

The generalization of Theorem 2.1 to the case with more than two subproblems can be stated as follows:

Theorem 3.1. *Given a column $\bar{x}^k$ to the k-th subproblem* (4) *and an upper bound $\bar{z}^{UB}$ on the optimum of the original problem* (1). *Column $\bar{x}^k$ is strongly redundant if one of the following conditions holds:*

(i) Problem (33) *is infeasible,*

(ii) for the optimum $z''(\bar{x}^k)$ of Problem (33) *it holds that*

$$\bar{z}^{UB} - c_1^T \bar{x}^k < z''(\bar{x}^k) \ .$$

Proof. Can be proven analogously to Theorem 2.1, using variants of Lemma 2.2 and Lemma 2.3 for multiple subproblems. □

Analogously to Theorem 2.1, Theorem 3.1 can be used to identify strongly redundant columns. Similar to the procedure explained in Section 2.2, the generation of such columns can then be avoided by using the dual solutions and dual rays of practical verification LP (33) to construct classical Benders cuts.

Let S^k be the set of all dual solutions and let T^k be the set of all dual rays of practical verification LP (33), i.e.,

$$S^k := \Big\{ w = \Big(w^0, (w^{k'} : k' \in K \setminus \{k\}) \Big) : (34\text{b}) - (34\text{d}) \Big\} \tag{35a}$$

$$T^k := \Big\{ w = \Big(w^0, (w^{k'} : k' \in K \setminus \{k\}) \Big) : (34\text{c}) - (34\text{d}), \tag{35b}$$

$$(w^0)^T A_{k'} + (w^{k'})^T D_{k'} \leq 0 \quad \forall k' \in K \backslash \{k\} \Big\} \ . \tag{35c}$$

Given a dual ray $\tilde{w} \in T^k$ of practical verification LP (33). Inequality (17) can be generalized as follows:

$$(\tilde{w}^0)^T (b - A_{k'} x^k) + \sum_{\substack{k' \in K \\ k' \neq k}} (\tilde{w}^{k'})^T d_{k'} \leq 0 \ . \tag{36}$$

Theorem 3.2. *Let $k \in K$ be fixed, and let $\tilde{w} \in T^k$ be a dual ray of Problem (33). Inequality (36) is valid for all solutions $\bar{x}^k \in X^k$ corresponding to columns that are not strongly redundant.*

Proof. Can be proven analogously to Theorem 2.3. □

Given a dual solution $\tilde{w} \in T^k$ of practical verification LP (33), we can analogously generalize Inequality (21) as follows:

$$c_{k'}^T x^{k'} + (\tilde{w}^0)^T (b - A_{k'} x^{k'}) + \sum_{\substack{k' \in K \\ k' \neq k}} (\tilde{w}^{k'})^T d_{k'} \leq \bar{z}^{UB} \ . \tag{37}$$

Theorem 3.3. *Let $k \in K$ be fixed, and let $\tilde{w} \in S^k$ be a dual solution of Problem (33). Furthermore, let $\bar{z}^{UB}$ be a primal bound for (1). Inequality (37) is valid for all solutions $\bar{x}^k \in X^k$ corresponding to columns that are not strongly redundant.*

Proof. Can be proven analogously to Theorem 2.4. □

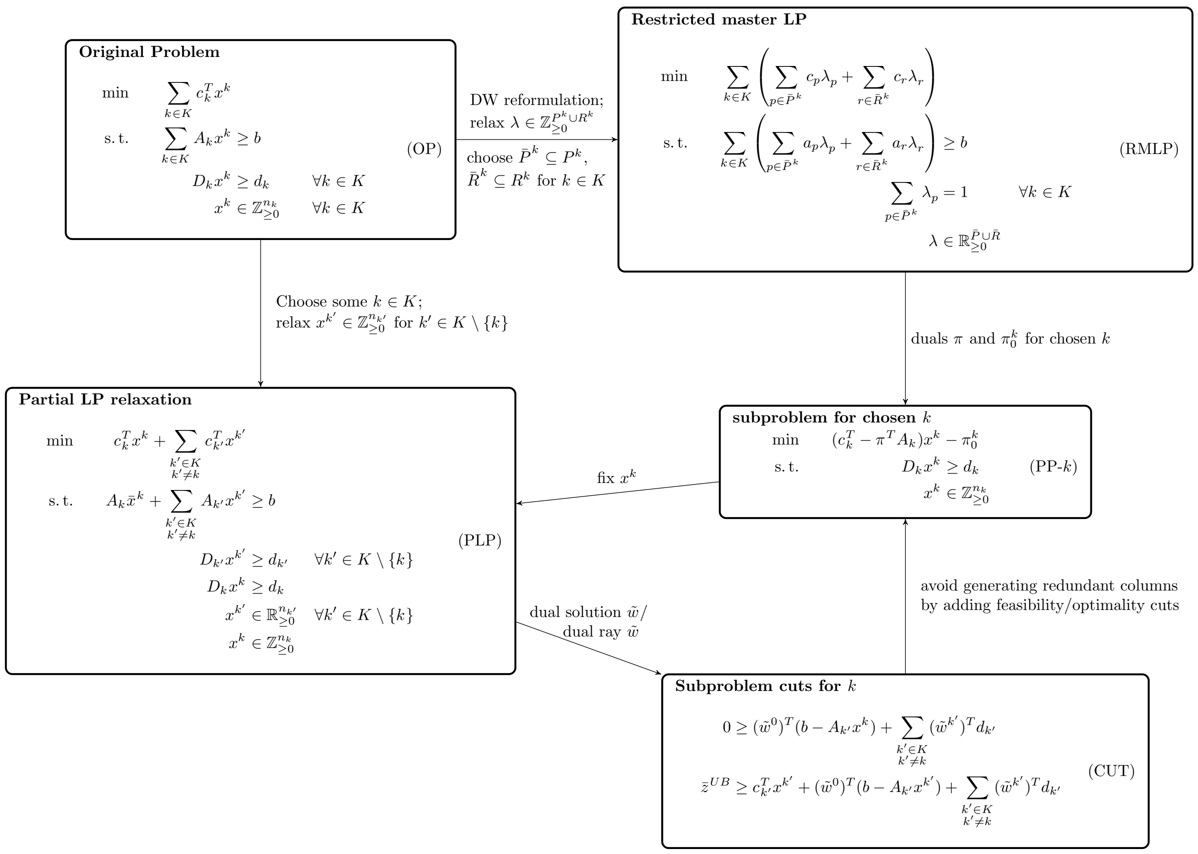
3.2 Mixed-Integer Programs

When applying the discretization form of Dantzig-Wolfe reformulation to a MIP, one obtains integrality constraints on sums of λ-variables [27]: For each integer x-variable, there exists an integrality constraint in the reformulated problem for the sum of λ-variables corresponding to subproblem solutions with the same solution value for this particular integer x-variable. Notice that the reformulated problem (3) for IPs contains integrality constraints for each individual λ-variable. Hence, Theorem 3.1 can be extended to MIPs by only fixing the integer x-variables of a given solution $\bar{x}^k$ in practical verification LP (33) and by adding the constraints $D_k x^k \geq d_k$ to practical verification LP (33).

4 Algorithm for Avoiding the Generation of Redundant Columns

Identifying and then, subsequently, avoiding the generation of redundant columns involves a modification to the pricing stage of the column generation algorithm. Briefly, the solutions found by the subproblem are checked for strong redundancy, using the methods previously described. If the solution induces a redundant column, then a subproblem cut is generated. Algorithm 1, presented in Section 4.1, describes the modified pricing iteration that includes the redundancy check based on Theorem 3.1. Note that we use the practical variant to generate subproblem cuts, presented in Section 2.3.

To better explain the connection between the many different mathematical programs introduced in the previous sections, a diagram of their relationship in the method for identifying redundant columns and avoiding their generation is given in Figure 1. The different mathematical programs are variants of problems that commonly arise in the context of Dantzig-Wolfe reformulation and Benders decomposition.



Original Problem
$\min \sum_{k\in K} c_k^T x^k$
$\text{s.t.} \sum_{k\in K} A_k x^k \geq b$
$D_k x^k \geq d_k \quad \forall k \in K$
$x^k \in \mathbb{Z}^{n_k}_{\geq 0} \quad \forall k \in K$
(OP)
DW reformulation; relax $\lambda \in \mathbb{Z}^{P^k \cup R^k}_{\geq 0}$
choose $\bar{P}^k \subseteq P^k$, $\bar{R}^k \subseteq R^k$ for $k \in K$
Restricted master LP
$\min \sum_{k\in K} \left(\sum_{p\in\bar{P}^k} c_p\lambda_p + \sum_{r\in\bar{R}^k} c_r\lambda_r \right)$
$\text{s.t.} \sum_{k\in K} \left(\sum_{p\in\bar{P}^k} a_p\lambda_p + \sum_{r\in\bar{R}^k} a_r\lambda_r \right) \geq b$
$\sum_{p\in\bar{P}^k} \lambda_p = 1 \quad \forall k \in K$
$\lambda \in \mathbb{R}^{\bar{P}\cup\bar{R}}_{\geq 0}$
(RMLP)
Choose some $k \in K$; relax $x^{k'} \in \mathbb{Z}^{n_{k'}}_{\geq 0}$ for $k' \in K \setminus \{k\}$
duals π and π_0^k for chosen k
Partial LP relaxation
$\min c_k^T x^k + \sum_{k'\in K, k'\neq k} c_{k'}^T x^{k'}$
$\text{s.t.} A_k\bar{x}^k + \sum_{k'\in K, k'\neq k} A_{k'} x^{k'} \geq b$
$D_{k'} x^{k'} \geq d_{k'} \quad \forall k' \in K \setminus \{k\}$
$D_k x^k \geq d_k$
$x^{k'} \in \mathbb{R}^{n_{k'}}_{\geq 0} \quad \forall k' \in K \setminus \{k\}$
$x^k \in \mathbb{Z}^{n_k}_{\geq 0}$
(PLP)
subproblem for chosen k
$\min (c_k^T - \pi^T A_k)x^k - \pi_0^k$
$\text{s.t.} D_k x^k \geq d_k$
$x^k \in \mathbb{Z}^{n_k}_{\geq 0}$
(PP-k)
fix x^k
avoid generating redundant columns by adding feasibility/optimality cuts
dual solution $\tilde{w}$/ dual ray $\tilde{w}$
Subproblem cuts for k
$0 \geq (\tilde{w}^0)^T(b - A_{k'}x^k) + \sum_{k'\in K, k'\neq k} (\tilde{w}^{k'})^T d_{k'}$
$\bar{z}^{UB} \geq c_{k'}^T x^{k'} + (\tilde{w}^0)^T(b - A_{k'}x^{k'}) + \sum_{k'\in K, k'\neq k} (\tilde{w}^{k'})^T d_{k'}$
(CUT)

Figure 1: An overview of various problems, and their relationship, used in the column generation algorithm and the generation of the subproblem cuts.

Given a solution $\bar{x}^k$ to the k-th subproblem for some $k \in K$, we can interpret solving practical verification LP (33) as solving the partial LP relaxation (PLP) of the original problem (OP) from Figure 1 with partially fixed solution $x^k = \bar{x}^k$. This gives a lower bound (which might be infinite if practical verification LP (33) is infeasible) on the optimum of the original problem with partially fixed solution $x^k = \bar{x}^k$. Given an upper bound on the optimum (or given that practical verification LP (33) is infeasible), we can generate a subproblem cut for the k-th pricing problem using a dual ray or a dual solution as depicted by (CUT) in Figure 1.

4.1 Modified Pricing Iteration

As in a regular column generation pricing iteration, the output of Algorithm 1 is a set of negative reduced cost columns. If no such columns exist, then the empty set is returned—indicating that the current solution to the restricted master LP is optimal for the master LP.

```
Data: Upper bound $\bar{z}^{UB}$, subproblems (4) (potentially including some subproblem cuts).
Result: Set $C$ of columns having negative reduced cost or $C = \emptyset$ if none exist.

$C := \emptyset$                    // set of columns that will be added to restricted master LP
$r_k := True \quad \forall k \in K$            // store if subproblem $k$ was refined/needs resolve
do
    $C_0 := \emptyset$                     // set of potential columns that will be added to $C$
    // (re)solve subproblems
    for $k \in K$ do
        if $r_k = True$ then
            Solve the $k$-th subproblem (4) and add columns $\bar{x}^k$ with negative reduced cost to $C_0$
            $r_k := False$                       // subproblem $k$ was solved
    end for
    // check columns in $C_0$ for redundancy
    for $\bar{x}^k \in C_0$ do
        $C_0 := C_0 \setminus \{\bar{x}^k\}$
        Solve practical verification LP (33) with optimum $z''$
        if practical verification LP (33) is infeasible then
            Construct feasibility cut and add it to the $k$-th subproblem (4)
            $r_k := True$                      // subproblem $k$ was refined
        else if $\bar{z}^{UB} - c_k^T \bar{x}^k < z''$ then
            Construct optimality cut and add it to the $k$-th subproblem (4)
            $r_k := True$                      // subproblem $k$ was refined
        else
            $C := C \cup \{\bar{x}^k\}$               // column $\bar{x}^k$ passed relaxed redundancy check
            $C_0 := C_0 \setminus \{\bar{x}^k\}$
    end for
    // exit loop if $C$ contains some columns or no subproblem was refined
while $C = \emptyset$ and there exists some $k \in K$ with $r_k = True$
return $C$
```

Algorithm 1: Pricing iteration with redundancy check.

The main steps of Algorithm 1 are as follows: First, subproblems (4) (potentially including some subproblem cuts) are solved. Then, all found columns $\bar{x}^k$ with negative reduced cost are checked for strong redundancy by solving the corresponding practical verification LP (33). If practical verification LP (33) is infeasible or the optimum is sufficiently large, we can construct a subproblem cut that is violated by $\bar{x}^k$ and add it to the k-th subproblem (4). Otherwise, we add $\bar{x}^k$ to the set of columns that will be added to the restricted master LP. Finally, we repeat this procedure: All subproblems that were changed by adding subproblem cuts in the previous round are resolved. If no subproblem cuts can be generated for the set of found columns with negative reduced cost, the algorithm terminates.

We remark that in each pricing iteration, the current upper bound $\bar{z}^{UB}$ can be used to strengthen the previously generated optimality cuts. Hence, the right hand side of the optimality cuts is updated whenever an improved upper bound is found.

5 Computational Results

We implemented the generation of subproblem cuts (see Algorithm 1) in the branch-price-and-cut solver GCG [13] that extends SCIP [1]. We used a development version of GCG 2.1.4, which is based on a development version of SCIP 5.0.1. Furthermore, we used CPLEX 12.7.1.0 as LP solver for the restricted master LP and as MIP solver for the subproblems.

Two different settings are used for these experiments: The first is *Default*, which is the default parameter settings for GCG, and the second is *Subcuts*, which is *Default* but with the generation of subproblem cuts—as described in Algorithm 1—enabled. In *Subcuts*, the subproblem cuts are only generated for the subproblems while solving the root node of the branch-and-price tree. For both settings the subproblems are first solved heuristically using a gap limit of 20% and a node limit of 1000 nodes. If no negative reduced cost columns are found using the heuristic pricing, the subproblems are then solved to optimality.

All computations were performed on a cluster consisting of Xeon L5630 Quad Core 2.13 GHz processors with 16 GB DDR3 RAM. The time limit used for all experiments is 3600 seconds.

5.1 Classical Instances

Classical problems for which Dantzig-Wolfe reformulation applies well were used for the initial computational experiments evaluating the potential of the subproblem cuts. Test instances were collected for bin packing, cutting stock, vertex coloring, capacitated p-median, generalized assignment, and single-source capacitated facility location problems. In this initial set of computational experiments on classical problems, (almost) no subproblem cuts were generated. While this is an undesired result, it is valuable in highlighting a limitation to the proposed methods and the subproblem cuts, and it also gives an insight into the nature of (strong) redundancy.

An explanation for the inability to generate subproblem cuts can be easily seen when considering the bin packing problem (similar arguments apply to the other classical problem classes). In the bin packing problem a set of n items is given, each having some positive weight a_i, $i \in \{1, \dots, n\}$, and a set of n bins, all having the same capacity. The goal is to pack the items into bins such that the capacity of the bins is not exceeded and the number of used bins is minimized. The classical textbook model reads

$$\begin{aligned}
\min \quad & \sum_{j=1}^{n} y_j && && \text{(38a)}\\
\text{s. t.} \quad & \sum_{j=1}^{n} x_{ij} = 1 && \forall i = 1, \dots, n && \text{(38b)}\\
& \sum_{i=1}^{n} a_{ij} x_{ij} \leq b \cdot y_j && \forall j = 1, \dots, n && \text{(38c)}\\
& x_{ij}, y_j \in \{0, 1\} && \forall i, j = 1, \dots, n \ , && \text{(38d)}
\end{aligned}$$

where x_{ij} for $i, j \in \{1, \dots, n\}$ equals 1 to indicate that item i is packed in bin j, and 0 otherwise. Further, y_j for $j \in \{1, \dots, n\}$ indicates whether bin j is used ($y_j = 1$) or not ($y_j = 0$). Constraints (38c) are knapsack constraints that ensure that the capacity of each bin is not exceeded and constraints (38b) are set partitioning constraints enforcing that each item is packed in exactly one bin. When Dantzig-Wolfe reformulation is applied to (38), the knapsack constraints (38c) are chosen as subproblem constraints, yielding one subproblem for each bin. Since the bins are identical, the subproblems can be aggregated.

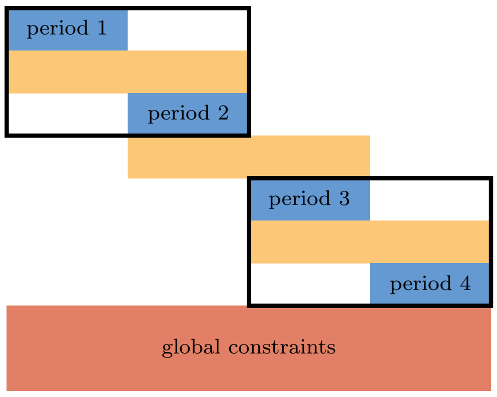
Suppose we are given a column, which corresponds to a packing of a bin, and we solve the corresponding practical verification LP (33) by choosing some bin k in the original problem as representative. Then practical verification LP (33) corresponds to fractionally packing all items that are not in the fixed packing into the other bins. Since finding such a packing is possible under mild conditions (the total weight of unpacked items should not exceed the total capacity of all other bins), practical verification LP (33) is usually feasible. Hence, Condition (i) of Theorem 3.1 is not satisfied and hence no feasibility subproblem cuts can be generated. Furthermore, the original LP relaxation of the bin packing problem is very weak, meaning that Condition (ii) of Theorem 3.1 is usually not satisfied; hence, (almost) no optimality subproblem cuts can be generated.

This explanation is particular to our methods but it may be inherent to the notion of (strong) redundancy. In the textbook models with classical decompositions that comprise many subproblems, a column generated from a single subproblem is very rarely not complemented to a feasible master problem solution with columns from other pricing problems. For objective functions that simply sum the master variables, there may be many optimal solutions, making it more unlikely for a column not to appear in any optimal solution. Even if it does not, it might not be generated in the subproblems and thus no subproblem cut would be generated.

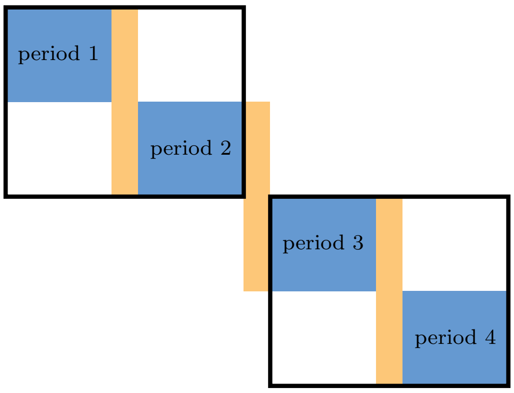
5.2 Temporal Decompositions

The computational experiments on classical instances highlighted that structure in the decomposition and dependency between the subproblems is necessary for the generation of subproblem cuts. A type of decomposition where such structure and dependency is exhibited is the temporal decomposition. Thus, to evaluate the potential of the subproblem cuts, the proposed methods have been applied when solving instances from lot sizing and unit commitment problems on which a temporal decomposition has been performed.

Temporal decompositions exploit a time-dependent structure between variable subsystems. There are two major types of temporal decompositions: period and horizon decompositions. A period decomposition is characterized by each subproblem being formed of a subsystem from a single time period. Alternatively, in horizon decompositions each subproblem comprises subsystems from multiple, consecutive time periods. Figure 2 illustrates the temporal structure of lot sizing [24] and unit commitment problems [12]. A period decomposition for unit commitment problems is obtained by choosing the constraints that only belong to one period (blue constraints in Figure 2a) as subproblem constraints and all other constraints (orange and red constraints in Figure 2a) as master constraints, resulting in one subproblem per period. A horizon decomposition with horizon k is obtained by choosing the constraints that belong to k consecutive periods as well as some related linking constraints as subproblem constraints. Particularly, the first k periods will form the first subproblem, the second k periods the second subproblem, etc., resulting in one subproblem for every consecutive k periods. Constraints that only link periods belonging to the same subproblem are also chosen as subproblem constraints (some of the orange constraints in Figure 2a). In the horizon decomposition with horizon 2 for the example illustrated in Figure 2a, the orange constraints linking the first and the second period as well as the ones linking the



(a) Unit commitment problems.



(b) Lot sizing problems.

Figure 2: Figure illustrating the temporal structure in the constraint matrix of the unit commitment and lot sizing problems. The horizon decomposition with horizon 2 is indicated by the black rectangles: Each black rectangle corresponds to a subproblem.

third and the fourth period are chosen as subproblem constraints, as indicated by the black rectangles. Analogously, by replacing linking constraints with linking variables, we can apply horizon decomposition to lot sizing problems, as depicted in Figure 2b.

We remark that there is a difference between horizon decompositions for unit commitment and lot sizing problems: In lot sizing problems, horizon decompositions are applied by introducing linking variables between different subproblems, whereas horizon decompositions for unit commitment problems introduce linking constraints instead.

A particular interest in temporal decomposition comes from recent successes reported when such a Dantzig-Wolfe reformulation is applied to lot sizing and unit commitment problems. Previously, temporal decomposition has been applied to single-level capacitated lot sizing problems with setup times [25] in order to obtain a *period decomposition* [6, 19]. More generally, temporal decompositions are applied to lot-sizing problems to obtain *horizon decompositions* [11]. As mentioned by de Araujo et al. [6], these decompositions can also be applied to multi-level capacitated lot sizing problems with setup times. Similar decompositions can also be applied to unit commitment problems. In particular Kim et al. [16] applied horizon decompositions (called *temporal decompositions* in their paper) to linearized unit commitment problems.

The test instances for the computational experiments presented in this paper consist of multi-level lot sizing problems instances collected from Tempelmeier and Derstroff [24] and the unit commitment instances collected from Frangioni [12]. For the multi-level lot sizing problems by Tempelmeier and Derstroff [24] we consider both period and horizon decompositions. In the computational experiments, the test set including the period decomposition is labeled ls-Derstroff-period. Since there are only 4 time periods, we restrict the horizon decompositions to have a horizon of 2. The corresponding test set is labeled ls-Derstroff-horizon2. For the unit commitment problem, we apply horizon decompositions with horizons 8 (uc-Frangioni-horizon8) and 12 (uc-Frangioni-horizon12) to linearized thermal unit commitment instances from Frangioni [12], which consist of 24 planning periods.

5.2.1 Example of Subproblem Cuts for Lot Sizing Problems

The proposed methods for generating subproblem cuts employs a Benders-like cut generating LP. As such, the generated cuts may take a form that does not have any practical meaning. However, for the Lot Sizing Problem, after applying a temporal decomposition, the cut generating LP produces subproblem cuts that have a practically meaningful form.

When considering temporal decompositions for lot sizing problems, each subproblem corresponds to some consecutive subset of periods and each column that is generated by such subproblem represents a possibly infeasible production plan for those periods. The subproblems are linked by demand or balance constraints. In the following, we will assume that period decomposition was applied, but similar subproblem cuts exist for horizon decompositions.

Suppose that the original lot sizing problem contains binary variables y_{it} for each product i and each period t with

$$y_{it} = \begin{cases} 1, & \text{if product } i \text{ is produced in period } t, \\ 0, & \text{else.} \end{cases}$$

These variables usually have setup times and costs. When solving practical verification LP (33) for a column (corresponding to a production plan for period t), some of these variables are set to 0, which can result in an infeasible problem. A subproblem feasibility cut for period t then forbids setting some of these variables to 0; for some subset I' of products the subproblem feasibility cut has the form

$$\sum_{i \in I'} y_{it} \geq 1 \ .$$

Similar subproblem cuts can be generated with horizon decompositions. For this type of decomposition, the cut is not associated with a single time period t, but a set of consecutive time periods $T_k \subseteq T$ that form a given horizon k.

5.2.2 Solving the Linear Programming Relaxation

Table 1 gives an overview of the computational results for solving the LP relaxation of the master problem (3), called the master LP. This table contains the following columns: the number of instances (ninst); the number of affected instances, i.e, instances in which subproblem cuts are generated (naff); the average number of generated subproblem cuts per affected instance (ncuts); the number of instances in which the master LP is solved within the time limit (nsol); the shifted geometric mean of master LP solving time in seconds with a shift of 1 (time); the shifted geometric mean of the number of column generation iterations for the master LP with a shift of 1 (iters); and the shifted geometric mean of the gap closed in comparison to the original LP relaxation, i.e., the LP relaxation of the original problem (1) (gapcl). The geometric mean displayed in the columns time, iters, and gapcl is computed over the instances where the master LP was solved by both settings in the given time limit.

Subproblem cuts are generated on the majority of the instances from all test sets (c.f. Table 1, Column naff). Furthermore, a relative large number of subproblem cuts are found on these instances (c.f. Table 1, Column ncuts). When comparing *Default* and *Subcuts*, it can be observed that the number of instances in which the master LP can be solved within the time limit is almost identical (c.f. Table 1, Column nsol). For most other instances, only a small increase in the solution time for the master LP is observed. The exceptions to this are the unit commitment instances using horizon decompositions with horizon 8, reporting a large increase in the solution time for the master LP. Finally, it can be observed that with *Subcuts* the number of pricing iterations increases on lot sizing instances from the Derstroff test set, whereas it decreases on all other test sets.

An important observation from Table 1 is that the addition of the subproblem cuts results in an

	overall			*Default*				*Subcuts*			
	ninst	naff	ncuts	nsol	time	iters	gapcl	nsol	time	iters	gapcl
ls-Derstroff-period	81	59	30.20	80	1.61	39.85	8.10	81	2.43	40.36	13.69
ls-Derstroff-horizon2	81	53	39.13	62	1.13	73.53	84.78	62	1.17	73.42	87.94
uc-Frangioni-horizon8	42	40	144.25	10	47.86	98.00	78.76	10	64.02	90.72	83.88
uc-Frangioni-horizon12	42	36	36.14	10	39.90	50.65	97.97	10	48.07	50.01	98.04

Table 1: Computational results when solving the master LP of the lot sizing and unit commitment instances using *Default* and *Subcuts*.

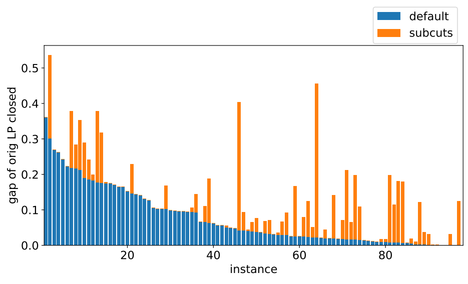
increase to the root node dual bound across many of the considered instances. For the Derstroff test sets, ls-Derstroff-period and ls-Derstroff-horizon2, an increase in the dual bound of 5.59% and 3.16%, respectively, is observed on average. A similar result is also shown for the uc-Frangioni-horizon8 and uc-Frangioni-horizon12 instances, reporting an average increase in the dual bound of 5.12% and 0.07%. These results demonstrate the ability of the subproblem cuts to improve the master LP dual bounds on structured integer programs.

It must be noted that the *Default* settings for GCG includes the domain propagation method of Gamrath and Lübbecke [13]. Thus, the results presented here demonstrate an improvement over the only other general method for avoiding the generation of redundant columns provided in the literature. This suggests that the subproblem cuts are stronger than the variable bounds derived from performing domain propagation on the original problem.

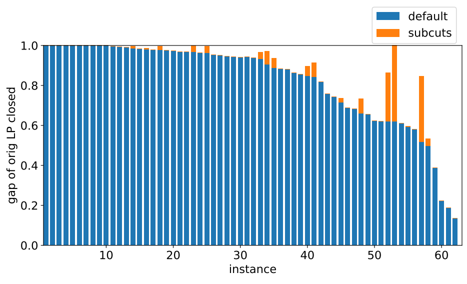
The methods proposed in this paper introduce extra work in the pricing stage of the column generation algorithm to avoid the generation of redundant columns. While an increase in the dual bound is achieved, this comes at the cost of an increased time to solve the root node of the restricted master LP. It can be seen in Table 1 that using Algorithm 1 in the pricing stage increases the time to solve the root node of the restricted master LP for all test sets. Unfortunately, this cost is unavoidable when generating subproblem cuts. Ideally, the extra work to increase the dual bound will aid in solving the restricted master LP to integer optimality, which is evaluated in Section 5.2.3.

Interestingly, the results show that only feasibility cuts are generated on all tested instances. Even if we provide the optimum z^* instead of an $\bar{z}^{UB}$ when generating the subproblem cuts. This could be explained by weak original LP relaxations and hence, "weak" practical verification LPs (33). In this setting, it appears that even by fixing a column from one pricing problem the objective function value of practical verification LP is never greater than the objective value of the integer optimal solution. This observation highlights a potential direction of future work to investigate alternative relaxations of verification IP to improve the strength of the subproblem cuts.

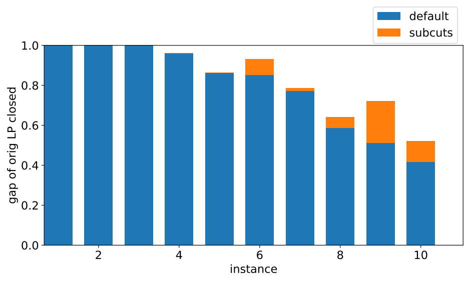
The improved dual bound on multi-level lot sizing and unit commitment instances is depicted in Figure 3. The y-axis shows the percentage of the gap of the original LP that was closed by the Dantzig-Wolfe reformulation and subproblem cuts, where a value of 1.0 indicates that the complete gap was closed. Except for the unit commitment instances using horizon decompositions of horizon 12, the



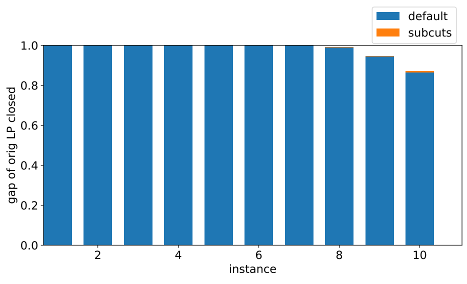
(a) ls-Derstroff-period



(b) ls-Derstroff-horizon2



(c) uc-Frangioni-horizon8



(d) uc-Frangioni-horizon12

Figure 3: The gap of the original LP relaxation that was closed by Dantzig-Wolfe reformulation (blue) and the additional gap that was closed by the generation of subproblem cuts (orange).

subproblem cuts close a large amount of the gap on several instances. Note that the gap that was closed varies considerably, even on instances of the same test set. An interesting observation from Figure 3 is that as the horizon period increases, the ability to improve the dual bound decreases. This could be explained by the same reason why subproblem cuts are most effective when a temporal decomposition is applied: An increased horizon period results in less subproblems, reducing the possibility of columns causing infeasibilities between subproblems. Considering the uc-Frangioni-horizon12 instances, the total number of time periods is 24, so horizons of 12 means that there are only 2 subproblems. As such, within each horizon period there will be no infeasibilities caused by the scheduling decisions between the time periods. The infeasibilities caused by preceding scheduling decisions can only occur between the two horizon periods—significantly limiting the number of subproblem cuts that can be generated, which is shown in Table 1. A similar result is observed for the ls-Derstroff-horizon2 instances, shown in Figure 3b, since there are only 4 time periods the horizon of 2 affords less opportunities to identify infeasibilities between the periods. Thus, the results show that subproblem cuts are most useful when there are many subproblems and the decomposition structure is such that fixing decisions from one subproblem causes infeasibilities in the others.

5.2.3 Solving the Integer Program

Table 2 presents the computational results for solving the master problem (3) using a branch-price-and-cut algorithm. Table 2 contains similar columns as Table 1 with the following differences: The Columns nsol and time correspond to solving the problems to integer optimality instead of solving only the LP

	overall		*Default*				*Subcuts*			
	ninst	naff	nsol	time	nds	gapcl	nsol	time	nds	gapcl
ls-Derstroff-period	81	59	56	72.50	865.69	8.10	56	68.08	793.79	13.69
ls-Derstroff-horizon2	81	53	58	13.62	35.06	84.78	58	9.91	18.97	87.94
uc-Frangioni-horizon8	42	40	6	95.15	6.56	78.76	6	79.11	4.52	83.88
uc-Frangioni-horizon12	42	36	8	31.86	1.32	97.97	8	34.59	1.32	98.04

Table 2: Computational results when solving the lot sizing and unit commitment instances by branch-and-price using *Default* and *Subcuts*.

relaxation, and Column nds specifies the geometric mean of the number of nodes in the branch-and-price tree on instances solved in both settings.

Comparing *Default* and *Subcuts*, we observe a general decrease in the run times from the use of the subproblem cuts. The largest absolute difference in the average run times is observed for the uc-Frangioni-horizon8 test set, with a 16.04 second average decrease, where the largest relative decrease (time(*Default*) − time(*Subcuts*)/time(*Default*)) is given by the ls-Derstroff-horizon2 test set (27.24%). While a general decrease in the run times is observed, there are still instances where the addition of the subproblem cuts has a negative effect. In particular, the uc-Frangioni-horizon12 test set exhibits an average increase of 2.73 seconds. An important observation is that while the generation of subproblems cuts may not always be effective, Algorithm 1 does not greatly affect the overall solving performance. However, when the subproblem cuts are effective, avoiding the generation of redundant columns can significantly improve the performance of the branch-and-price algorithm.

The benefit for improving the root node dual bound with the addition of subproblem cuts is shown by a decrease in the number of branch-and-bound nodes to solve the instances. A large decrease is observed for both lot sizing and uc-Frangioni-horizon8 test sets, with the largest relative decrease of 45.89% produced by the ls-Derstroff-horizon2 instances. This result demonstrates that the tighter root node relaxation achieved by eliminating redundant columns can have an overall positive effect on the performance of the branch-and-price algorithm.

5.2.4 Impact of Subproblem Cuts on the Difficulty of Subproblems

In this section, we turn our attention to the computational impact of generating subproblem cuts. Since more constraints are added to the subproblems, it is expected that these problems become more difficult to solve. However, the addition of the subproblem cuts aims to eliminate generators that correspond to redundant columns. As such, it may be possible to observe a decrease in the number of pricing iterations. Average results for the complete test sets will be presented along with details for specific instances to better demonstrate the effect that the addition of subproblem cuts has on the pricing of new columns.

In Table 3 we display information on solution times of the subproblems. Table 3 contains the columns: the geometric mean of the overall pricing times per instance (ptime), the number of column generation

	overall		*Default*			*Subcuts*		
	ninst	naff	ptime	piters	avgptime	ptime	piters	avgptime
ls-Derstroff-period	81	59	1.3843	963.7515	0.0009	2.2779	971.8153	0.0014
ls-Derstroff-horizon2	81	53	1.0380	326.7728	0.0034	1.0851	327.7981	0.0035
uc-Frangioni-horizon8	42	42	66.1607	1049.2352	0.0870	67.1152	975.5090	0.0882
uc-Frangioni-horizon12	42	42	37.0855	184.6146	0.2287	46.5918	174.6371	0.3226

Table 3: subproblem statistics when solving the master LP of the lot sizing and unit commitment instances using *Default* and *Subcuts*.

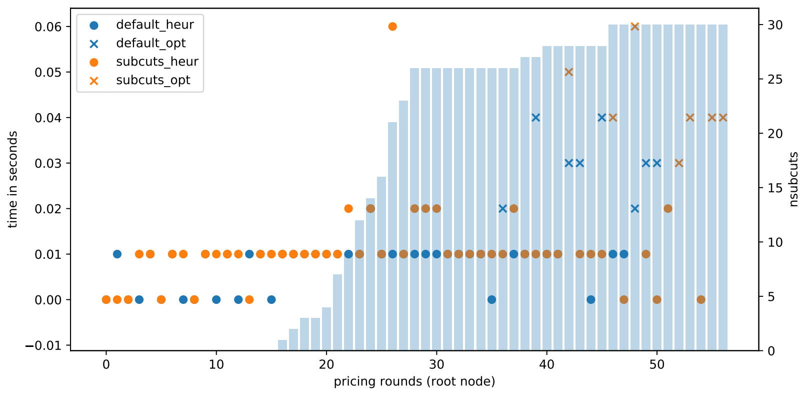
pricing iterations (piters) and of the average pricing time per instance (avgptime).

We observe in Table 3 the generation of subproblem cuts only results in a small increase in the average solving time of the subproblems. However, the magnitude of the increase in the subproblem solving time depends greatly on the test set. Overall, the results in Table 3 suggests that the subproblem cuts do not have a significant effect on the difficulty of the subproblems.

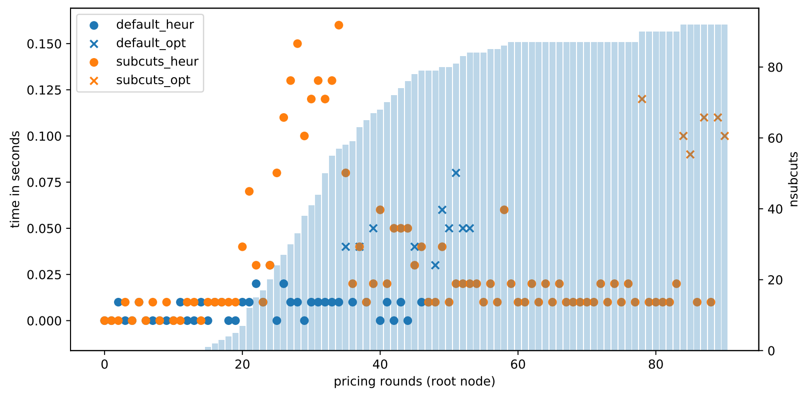
The exception to the preceding results is the uc-Frangioni-horizon12 test set, where a large increase in the pricing time is observed. This result could be explained by the fact that the horizon of 12 time periods already results in a more difficult subproblem compared to a horizon of 8 time periods, with an average of 0.2287 and 0.087 seconds per pricing iteration respectively. Thus, the subproblem cuts exacerbate the difficulty in solving the subproblems. Also, the subproblem cuts will add further linking to the time periods within the selected horizon. As a result, these inequalities can destroy the separability within the subproblems and affect the performance of the MIP solver. These results indicate that the generation of subproblem cuts is most advantageous when the subproblem is not too difficult. Identifying such limits of difficulty is a topic of future work.

In Figure 4, we depict the solution times of the subproblems with and without subproblem cuts (the left-hand axis). Additionally, we present, on the right-hand axis, the number of generated subproblem cuts. The subproblem solution times in Figure 4 is given by the dots and crosses for the heuristic and exact pricing methods. Also, the columns in Figure 4 cumulatively show the number of subproblem cuts generated. Note that the number of pricing rounds can differ between the *Default* and *Subcuts* settings.

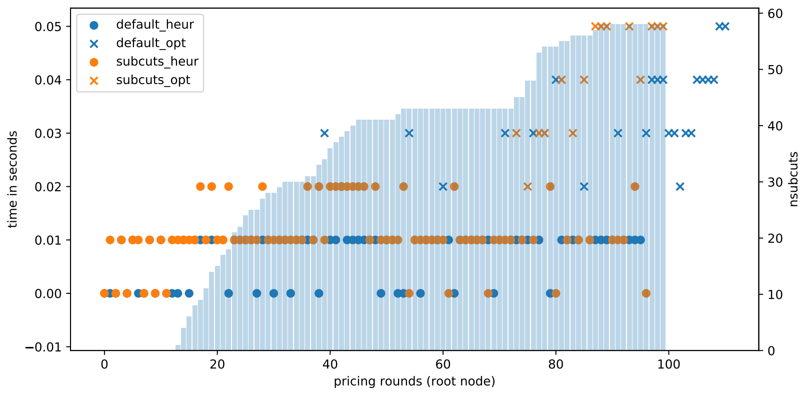
The results presented in Figure 4 verify the average results from Table 3 showing that there is little difference between the subproblem solving times for *Default* and *Subcuts* in most iterations. The biggest difference between the two settings is that after generating subproblem cuts in the beginning of the column generation algorithm, for some instances the subproblem solving times can increase significantly (c.f. Figure 4a and 4b), but then decrease to the solving times exhibited by *Default*. This behavior can be explained by the importance of the subproblem cuts in the generation of non-redundant columns. After the addition of some subproblem cuts, these constraints can become tight in an optimal solution to the subproblems. Since these constraints are typically more numerically difficult than the original problem constraints, and the fact that they link periods within each horizon, their existence affects the run times for the subproblem. After the initial generation of subproblem cuts, less columns are found



(a) G0241221-horizon2 (Derstroff)



(b) K0014535-horizon2 (Derstroff)



(c) K5017113-horizon2 (Derstroff)

Figure 4: Solution times (primary y-axis) and number of subproblem cuts (secondary y-axis) are depicted in each pricing iteration (x-axis) on three lot sizing instances. Furthermore, we indicate whether the subproblems were solved heuristically ({default,subcuts}_heur) or optimally ({default,subcuts}_opt).

to be redundant and fewer subproblem cuts are tight, leading to a decrease in the time for each pricing iteration. Additionally, solution times can be large when solving the subproblems to optimality, which is often the case in the last pricing iterations, if subproblem cuts have been added. However, this increase in the subproblem solving times at the end of computation is similar for both *Default* and *Subcuts*. Overall, the results show that the addition of subproblem cuts can improve the dual bound with little increase in the solving times of the column generation subproblems.

6 Conclusions

Our methods for identifying redundant columns and avoiding their generation extends the work of Vanderbeck and Savelsbergh [27] and Gamrath and Lübbecke [13]. Instead of only tightening variable bounds in the subproblems, this work, to the best of the authors' knowledge, is the first to propose a more general approach that uses Benders-type inequalities to avoid the generation of a subset of redundant columns using so-called *subproblem cuts*. Subproblem cuts are generated by exploiting classical Benders cuts, which is a novel, interesting integration of Benders decomposition and Dantzig-Wolfe reformulation.

Although generating subproblem cuts, which are based on information from the master constraints (i.e., on *global* information), is contrary to the decomposition principle, these cuts do not greatly increase the difficulty of the subproblem. The main benefit of generating subproblem cuts is a stronger relaxation yielding tighter dual bounds. Initial experiments showed that it was not possible to generate subproblem cuts on classical problem classes where Dantzig-Wolfe reformulation applies well. Our investigations identified that the generation and addition of subproblem cuts is most efficacious for problems exhibiting structure, in particular time-dependent structure. The computational experiments show that the addition of subproblem cuts can achieve tighter dual bounds and an improved performance for problems where a temporal decomposition is performed.

While the generation of subproblem cuts achieves an improvement in the dual bound, this improvement is not as great as expected. There are two main explanations for this result. First, the proposed methods to generate subproblem cuts are based on the LP relaxation of the original problem, which can be much weaker than the master LP. This could be a major reason why no optimality cuts are generated on the considered test instances. The second explanation is that we solve an individual LP for each potentially new column, which can be time consuming if a large number of columns are generated. If we would base the generation of subproblem cuts on a different, stronger relaxation (in combination with Lemma 2.3), generating subproblem cuts would be even more time consuming, but we could obtain even stronger dual bounds.

Given the potential of the proposed methods for avoiding the generation of redundant columns, a promising and interesting area of future research is the investigation of alternative relaxations of practical verification LP that could improve the computational performance. One direction of future research is to use the master LP, instead of the original LP, in combination with Lemma 2.3 in order to generate stronger subproblem cuts. In this case one would solve the practical verification LP using column generation, which may not be very efficient. Furthermore, this raises the question on how to handle columns that are generated while checking redundancy: should redundancy of these columns be checked as well or should we just disregard these columns in the master LP? Finally, in the presented methods, only classical Benders optimality and feasibility cuts are added to the subproblem. Another direction of future research is to investigate the use of Benders decomposition enhancement techniques while generating subproblem cuts.

Acknowledgments

The authors would like to thank the anonymous reviewers and editors for their helpful comments that greatly improved the quality of this paper. SJM was supported by the Engineering and Physical Sciences Research Council (EPSRC) [EP/P003060/1].

References

[1] T. Achterberg. SCIP: Solving constraint integer programs. *Mathematical Programming Computation*, 1(1):1–41, 2009.

[2] R. Baldacci, N. Christofides, and A. Mingozzi. An exact algorithm for the vehicle routing problem based on the set partitioning formulation with additional cuts. *Mathematical Programming*, 115(2):351–385, 2008.

[3] C. Barnhart, E. Johnson, G. Nemhauser, M. Savelsbergh, and P. Vance. Branch-and-price: column generation for solving huge integer programs. *Operations Research*, 46(3):316–329, 1998.

[4] J. E. Beasley. A Lagrangian heuristic for set-covering problems. *Naval Research Logistics*, 37(1):151–164, 1990.

[5] S. Ceria, P. Nobili, and A. Sassano. A Lagrangian-based heuristic for large-scale set covering problems. *Mathematical Programming*, 81(2):215–228, 1998.

[6] S. A. de Araujo, B. D. Reyck, Z. Degraeve, I. Fragkos, and R. Jans. Period decompositions for the capacitated lot sizing problem with setup times. *INFORMS Journal on Computing*, 27(3):431–448, 2015.

[7] J. Desrosiers and M. Lübbecke. A primer in column generation. In G. Desaulniers, J. Desrosiers, and M. Solomon, editors, *Column Generation*, pages 1–32. Springer, Berlin, 2005.

[8] T. Fahle, U. Junker, S. E. Karisch, N. Kohl, M. Sellmann, and B. Vaaben. Constraint programming based column generation for crew assignment. *Journal of Heuristics*, 8(1):59–81, 2002.

[9] T. Fahle and M. Sellmann. Cost based filtering for the constrained knapsack problem. *Annals of Operations Research*, 115(1):73–93, 2002.

[10] J. Farkas. Theorie der einfachen Ungleichungen. *Journal für die Reine und Angewandte Mathematik*, 124:1–27, 1902.

[11] I. Fragkos, Z. Degraeve, and B. D. Reyck. A horizon decomposition approach for the capacitated lot-sizing problem with setup times. *INFORMS Journal on Computing*, 28(3):465–482, 2016.

[12] A. Frangioni, C. Gentile, and F. Lacalandra. Tighter approximated MILP formulations for unit commitment problems. *IEEE Transactions on Power Systems*, 24(1):105–113, 2009.

[13] G. Gamrath and M. E. Lübbecke. Experiments with a generic Dantzig-Wolfe decomposition for integer programs. In P. Festa, editor, *Experimental Algorithms*, Lecture Notes in Computer Science, pages 239–252, Berlin, 2010. Springer Berlin Heidelberg.

[14] A. Hadjar, O. Marcotte, and F. Soumis. A branch-and-cut algorithm for the multiple depot vehicle scheduling problem. *Operations Research*, 54(1):130–149, 2006.

[15] S. Irnich, G. Desaulniers, J. Desrosiers, and A. Hadjar. Path-reduced costs for eliminating arcs in routing and scheduling. *INFORMS Journal on Computing*, 22(2):297–313, 2010.

[16] K. Kim, A. Botterud, and F. Qiu. Temporal decomposition for improved unit commitment in power system production cost modeling. *IEEE Transactions on Power Systems*, 33(5):5276–5287, 2018.

[17] M. E. Lübbecke and J. Desrosiers. Selected topics in column generation. *Operations Research*, 53(6):1007–1023, 2005.

[18] G. L. Nemhauser and L. A. Wolsey. *Integer and Combinatorial Optimization.* Wiley Interscience Series in Discrete Mathematics and Optimization. Wiley, 1988.

[19] C. M. O. Pimentel, F. P. e Alvelos, and J. M. V. de Carvalho. Comparing Dantzig-Wolfe decompositions and branch-and-price algorithms for the multi-item capacitated lot sizing problem. *Optimization Methods and Software*, 25(2):299–319, 2010.

[20] E. Rönnberg and T. Larsson. Column generation in the integral simplex method. *European Journal of Operational Research*, 192(1):333–342, 2009.

[21] E. Rönnberg and T. Larsson. An integer optimality condition for column generation on zero-one linear programs. *Discrete Optimization*, 31:79–92, 2019.

[22] S. Rosat, I. Elhallaoui, F. Soumis, and A. Lodi. Integral simplex using decomposition with primal cuts. In J. Gudmundsson and J. Katajainen, editors, *Experimental Algorithms*, pages 22–33, Cham, 2014. Springer International Publishing.

[23] M. M. Sol. *Column Generation Techniques for Pickup and Delivery Problems*. PhD thesis, Technische Universiteit Eindhoven, Eindhoven, 1994.

[24] H. Tempelmeier and M. Derstroff. A Lagrangean-based heuristic for dynamic multilevel multiitem constrained lotsizing with setup times. *Management Science*, 42(5):738–757, 1996.

[25] W. W. Trigeiro, L. J. Thomas, and J. O. McClain. Capacitated lot sizing with setup times. *Management Science*, 35(3):353–366, 1989.

[26] F. Vanderbeck. On Dantzig-Wolfe decomposition in integer programming and ways to perform branching in a branch-and-price algorithm. *Operations Research*, 48(1):111–128, 2000.

[27] F. Vanderbeck and M. W. P. Savelsbergh. A generic view of Dantzig-Wolfe decomposition in mixed integer programming. *Operations Research Letters*, 34(3):296–306, 2006.

[28] A. Zaghrouti, F. Soumis, and I. El Hallaoui. Integral simplex using decomposition for the set partitioning problem. *Operations Research*, 62(2):435–449, 2014.