UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO

MARCELO WUTTIG FRISKE

A Branch-and-Price Algorithm for a Compressor Scheduling Problem

Thesis presented in partial fulfillment
of the requirements for the degree of
Master of Computer Science

Advisor: Profa. Dr. Luciana Salete Buriol
Coadvisor: Prof. Dr. Eduardo Camponogara

Porto Alegre
March 2016

CIP — CATALOGING-IN-PUBLICATION

Friske, Marcelo Wuttig

A Branch-and-Price Algorithm for a Compressor Scheduling Problem / Marcelo Wuttig Friske. – Porto Alegre: PPGC da UFRGS, 2016.

58 f.: il.

Thesis (Master) – Universidade Federal do Rio Grande do Sul. Programa de Pós-Graduação em Computação, Porto Alegre, BR–RS, 2016. Advisor: Luciana Salete Buriol; Coadvisor: Eduardo Camponogara.

1. Compressor scheduling problem. 2. Branch-and-price. 3. Column generation. 4. Piecewise-linear formulation. I. Buriol, Luciana Salete. II. Camponogara, Eduardo. III. Título.

ACKNOWLEDGMENTS

I would like to thank my parents Eldevir and Rosane for the opportunities that they give me since I was born, and I thank my lovely wife Mariana for the unconditional support of my decisions. I thank my aunt Norma that accepted me as her guest over two years. I am very grateful to my advisor Luciana and my coadvisor Eduardo for their support during my masters. Lastly, but not last, I thank my laboratory colleagues Árton, Fernando, Alex, André, Leonardo B, Leonardo M, Paulo, Germano, Tadeu, Henrique, Renato, Toni, Carlo and Victoria for the experience exchange and built friendships.

ABSTRACT

This work presents the study and application of a branch-and-price algorithm for solving a compressor scheduling problem. The problem is related to oil production and consists of defining a set of compressors to be activated, supplying the gas-lift demand of a set of wells and minimizing the associated costs. The problem has a non-convex objective function, to which a piecewise-linear formulation has been proposed. This dissertation proposes a column generation approach based on the Dantzig-Wolfe decomposition, which achieves tighter lower bounds than the straightforward linear relaxation of the piecewise-linear formulation. The column generation was embedded in a branch-and-price algorithm and further compared with CPLEX, obtaining optimal solutions in lesser time for a set of instances. Further, the branch-and-price algorithm can find better feasible solutions for large instances under a limited processing time.

Keywords: Compressor scheduling problem. branch-and-price. column generation. piecewise-linear formulation.

A Branch-and-Price Algorithm for a Compressor Scheduling Problem

RESUMO

O presente trabalho realiza o estudo e aplicação de um algoritmo de branch-and-price para a resolução de um problema de escalonamento de compressores. O problema é ligado à produção petrolífera, o qual consiste em definir um conjunto de compressores a serem ativados para fornecer gas de elevação a um conjunto de poços, atendendo toda demanda e minimizando os custos envolvidos. O problema é caracterizado por uma função objetivo não-convexa que é linearizada por partes de forma a ser formulada como um problema de programação inteira mista. A abordagem de geração de colunas é baseada na decomposição de Dantzig-Wolfe e apresenta melhores limitantes inferiores em relação à relaxação linear da formulação compacta. O branch-and-price é comparado ao *solver* CPLEX, sendo capaz de encontrar a solução ótima em menor tempo para um conjunto de instâncias, bem como melhores soluções factíveis para instâncias maiores em um período de tempo limitado.

Palavras-chave: Problema de Escalonamento de Compressores, Algorítmos Exatos, Geração de Colunas, Formulação Linear por Partes.

LIST OF ABBREVIATIONS AND ACRONYMS

B&P	Branch-and-price;
BKV	Best-known value;
CFLP	Capacitated Facility Location Problem;
CG	Column Generation;
CSP	Compressor Scheduling Problem;
GLPC	Gas-Lift Performance Curve;
MIP	Mixed Integer Problem;
MP	Master problem;
RHS	Right-hand side;
RMP	Restricted master problem;
SSCFLP	Single-Source Capacitated Facility Location Problem;

LIST OF FIGURES

LIST OF TABLES

CONTENTS

1 INTRODUCTION

The advance of the technologies and the growing access of the population to it demands energy. The energy is necessary for all stages of any process that involves product manufacturing, from the extraction of raw materials to the delivery to costumers. One of the most important sources of energy is oil. Its derivatives are used as fuel for most transportation facilities, besides serving as feedstock for the frabrication of other products. Due to this, huge investments in research are made in the petroleum area, reaching many knowledge areas such as geology, chemistry, physics, engineering, mathematics, and others. Researchers seek to improve all steps of the petroleum supply chain and may involve the use of operational research (OR) to maximize or minimize an objective, *e.g.* maximize the daily production or minimize the cost of production per barrel.

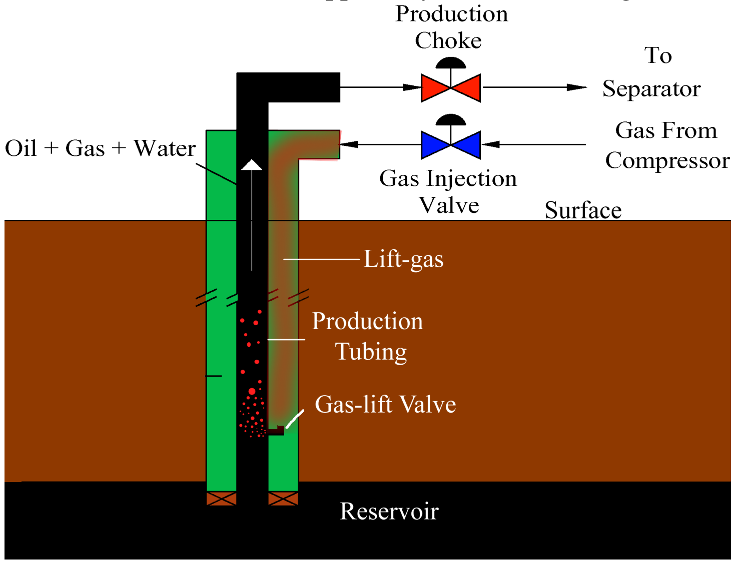
During the production phase in an oil field, after some time of extraction of the hydrocarbons (blend of oil, gas and water), the internal pressure of the wells may not be sufficient to lift these hydrocarbons to the surface or obtain profitable production levels. In this case, some artificial method as the continuous gas-lift is needed. Figure 1.1 illustrates the operation in a well supplied by gas-lift. In this technique, compressors produce high-pressure natural gas that is injected at the bottom of the production tubing at a certain rate and pressure entering in this tube through valves. The gas blends with the hydrocarbons, making it less dense due to the gasification, reducing the weight of the fluid column and then lifting the blend to the surface.

The wells must receive lift-gas at a certain rate and pressure to induce ideal production levels. Compressors have different capacities, installation and operation costs. From these issues, Camponogara, Castro e Plucenio (2007) defined and proposed a formulation for the Compressor Scheduling Problem (CSP). The CSP is a combinatorial problem which aims to minimize the costs involving the continuous gas-lift, define compressors to be installed (or activated) and the wells that will be supplied by each of these compressors.

The formulation of CSP is an extension of the Capacitated Facility Location Problem (CFLP), which is a well-studied combinatorial optimization problem. The CFLP is being classified as an economic decision problem applied in real situations, such as communications networks, locations of warehouses, logistics and others. The CFLP consists in defining locations to install or build facilities to meet the demand of consumers or clients and minimizing the installation costs and the service costs between facilities and clients.

The CSP and the CFLP can be formulated as a Mixed Integer Linear Problem (MILP) and as an Integer Problem (IP), respectively. The first uses continuous and integer variables,

Figure 1.1 – Illustration of well supplied by the continuous gas-lift technique



Source: Camponogara e Nakashima (2006) and adapted by the author (2016).

and the second uses only just integer variables. The exact solution for this type of problem can be found by techniques that combine linear problems algorithms as the Simplex, and methods based on simple ideas such as Branch-and-Bound (B&B) (CONFORTI; CORNUÉJOLS; ZAMBELLI, 2014).

Depending on the characteristics of a MILP or a IP, the linear relaxation provides weak bounds and can compromise the performance of B&B algorithms. For this reason, the Dantzig-Wolfe decomposition (DANTZIG; WOLFE, 1960) can be used to tighten these bounds. It consists in reformulating the problem in a model with a huge number of variables, where each variable defines a partial solution for the original problem. A Column Generation (CG) method to solve the linear relaxation of the decomposition was proposed by Camponogara e Plucenio (2008) for the CSP, presenting the decomposition and an illustrative example.

When an integer solution is provided by the CG, the solution is also optimal for the original problem. Otherwise, it is a relaxed solution which can be used in the Branch-and-Price (B&P) approach, *i.e.* the CG is embedded in a Branch-and-Bound framework, solving the column generation in each node of the enumeration tree.

This work presents a Column Generation for the CSP based on a revised model (CAMPONOGARA; NAZARI; MENESES, 2012), comparing the CG lower bounds with the relaxation of the revised model. Moreover, the work presents a Branch-and-Price algorithm for the CSP, comparing the computational results with the commercial solver CPLEX.

The remainder of this work is organized as follow: Chapter 2 presents a literature review. Chapter 3 describes the Compressor Scheduling Problem and its characteristics. Chapter 4 explains the column generation approach for the CSP. Chapter 5 presents the B&P technique used for solving the Compressor Scheduling Problem. Chapter 6 shows and compares the results obtained with the CG and B&P algorithm for the CSP. Finally, Chapter 7 presents the conclusion of this work and suggests future works that could be explored to solve the CSP.

2 RELATED WORK

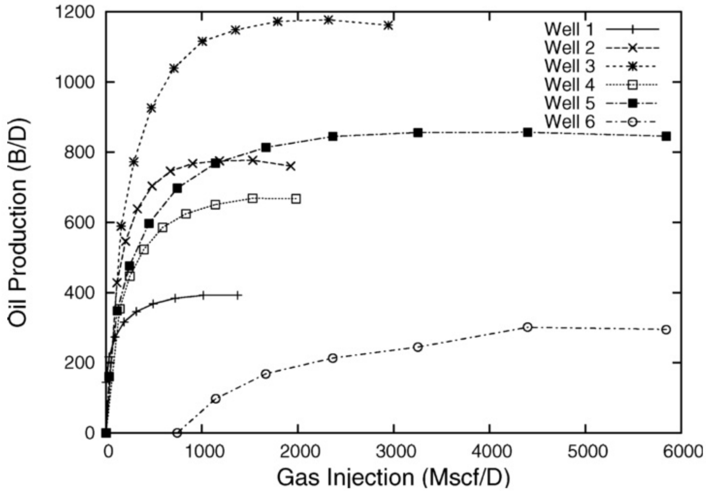
This chapter presents the main base concepts and the literature review related to the dissertation. In Section 2.1 we show some works whose objective is to optimize the use of the gas-lift technique, including the works related to the Compressor Scheduling Problem. The Facility Location Problem is presented in Section 2.2, with the formulation of a variation called Single-Source Capacitated Facility Location Problem, and approaches for solving it exactly and heuristically. Furthermore, some approaches for piecewise-linear formulations of nonlinear functions such as found in CSP are presented in Section 2.3. Finally, in Section 2.4, the Column Generation and Branch-and-Price methods are explained.

2.1 Continuous gas-lift

The continuous gas-lift technique, or simply gas-lift, is a preferable technique used due to its robustness, its relatively low installation and maintenance costs, and its wide range of operating conditions (CAMPONOGARA, 2006; HAMEDI; KHAMEHCHI, 2012). For this reason, we can find many works in the literature that aim to maximize the production and the profits using this technique. The gas-lift technique presents some characteristics which should be considered when used in the oil extraction. Each well has a gas-lift performance curve (GLPC) that can be described by a nonlinear function production vs. gas injection. Figure 2.1 shows the GLPC for a set of five wells.

The GLPC gives the maximum production of oil barrels per day (B/D) as a function of the injected gas (Mscf/d - thousand standard cubic feet gas per day) in the well. However, the GLPC does not necessarily provide the maximum profit in the production because the gas prices and compressing costs are not considered. It might be more profitable to produce less oil paying less for the gas-lift. Another case arises in an oil field with limited quantities of compressed gas, in which it can be more profitable to produce less oil in many wells than reach the maximum production of some wells. Based on these issues, Kanu, Mach e Brown (1981) present an economic slope method to deal with excessive usage of gas when it is unlimited and propose a method to allocate the gas to a group of six wells in a scenario with limited gas. Therein, the objective is to maximize the profit of the production, defining the gas-lift injection for a set of wells, based on the costs of compressed gas and the GLPC of the wells. The work of Nishikiori et al. (1989) uses a quasi-Newton method to solve the nonlinear optimization problem with

Figure 2.1 – Gas-lift performance curve examples



Source: (MISENER; GOUNARIS; FLOUDAS, 2009).

gradient projection for the gas-lift allocation optimization. Buitrago, Rodrigues e Espin (1996) proposed a multi-start algorithm that combines stochasticity and a heuristic calculation of a descent direction to determine the optimal gas-lift injection to maximize the production of a set of wells. To improve the performance of the gas-lift allocation, the work of Misener, Gounaris e Floudas (2009) compares four piecewise-linear formulations to solve the problem. The results showed that the method proposed by Keha, Farias e Nemhauser (2004) achieves best results regarding computational time. The work of Camponogara (2006) extends the gas allocation problem, incorporating logical constraints on the gas pipelines. They propose a dynamic programming algorithm to solve the problem approximately. In Khishvand e Khamehchi (2012), pressure drops are considered as constraints for the model, and a risk function for the oil and compressed gas prices is used, showing that it can influence operational parameters as the gas injection.

Another characteristic of the oil fields that use gas-lift is the need of a recovery plan for oil wells before the extraction. This plan establishes the gas rates and pressures along the lifespan of the wells to reach maximum production (CAMPONOGARA; CASTRO; PLUCENIO, 2006). However, these rates and pressures are overestimated due unpredicted events as compressor failures or dynamic changes in the reservoir conditions. If a compressor compress more gas than the total demand of wells that it supplies, it becomes necessary to or export or burn the gas excess (flaring). Moreover, if a compressor needs to reduce its capacity, there is a loss of energy, and consequently the production cost increases. These questions lead to the problem of allocating a set of compressors to meet the demands from a set of wells. A first formulation for the problem was proposed by Camponogara, Castro e Plucenio (2006), based on the Facility Location Problem, where compressors can be considered facilities and the wells are customers. The authors propose an algorithm based on dynamic programming to solve the problem. Camponogara, Castro e Plucenio (2007) proposed a formulation of the Compressor Scheduling Problem, which consists of defining a compressor gas-lift set to be activated, meeting the demand of gas rate and pressure of all oil wells and defining the gas rate produced by the compressors. Assuming that there is sufficient gas-lift for all wells, the objective is to minimize the compressor installation and operating costs and the pipeline costs between compressors and wells. The authors propose a piecewise-linear formulation to solve the problem using the CPLEX solver and valid inequalities to reduce the search space. Some experiments tests concluded that large instances need cutting planes for best performance.

Camponogara e Plucenio (2008) proposed a Dantzig-Wolfe decomposition for the CSP model presented in Camponogara, Castro e Plucenio (2007). A column generation procedure was presented with an illustrative example, but without experimental results. Camponogara et al. (2011) show that CSP is NP-Hard and proposes two families of valid inequalities for the problem. A two-phase method was proposed, consisting in generating cuts to improve the formulation and then solving the resulting problem with CPLEX. The results showed that cover-based cuts reduced the computational time and memory.

Camponogara, Nazari e Meneses (2012) presents a revised formulation for the previous CSP model. They replace the nonlinear constraint with a family of linear inequalities, reducing the resolution time ten-fold. Two formulations for the piecewise-linear formulation were tested using the generation of cutting planes to analyze the speed in the resolution. The results showed that valid inequalities improved time resolution in large instances, but did not impact as much as in the previous formulation.

Friske, Buriol e Camponogara (2015) presents a column generation for the CSP based on the revised model of (CAMPONOGARA; NAZARI; MENESES, 2012). The results show that the method can produce tighter lower bounds than the simple relaxation of the compact model.

This work proposes some improvements for the column generation procedure and a branch-and-price approach to solve the CSP exactly, comparing the results with the state-of-the-art solver CPLEX.

2.2 The Facility Location Problem

The Facility Location Problem (FLP) is a well-studied combinatorial optimization problem which basically involves a set of facilities that can be installed and a set of clients that must be serviced. The objective is to define which facilities will be installed or constructed to supply the demands of clients, such that the involved costs are minimized. Because of its generality, several studies have appeared in the literature during the last decades and many variations were proposed to apply the model in diverse fields, such as telecommunications, transports, and others (for examples, see Contreras, Díaz e Fernández (2010), Klose e Drexl (2005) and Eiselt e Marianov (2011)).

A variation of the FLP known as Single-Source Capacitated Facility Location Problem (SSCFLP) consists of a problem with two particularities. The first, “Single Source” requires that each costumer must be supplied by only one facility, and the second, “Capacitated” considers that the costumers have demands to be supplied and facilities have capacities that should be respected. We define a set $N = \{1, \ldots, n\}$ of facilities and a set $M = \{1, \ldots, m\}$ of clients. The SSCFLP can be formulated as follows:

$$\min \quad \sum_{j \in N} c_j y_j + \sum_{i \in M} \sum_{j \in N} c_{ij} x_{ij} \tag{2.1a}$$

$$\text{s.t.} \; : \quad x_{ij} \leq y_j, \quad i \in M, \; j \in N \tag{2.1b}$$

$$\sum_{j \in N} x_{ij} = 1, \quad i \in M \tag{2.1c}$$

$$\sum_{i \in M} d_i x_{ij} \leq b_j, \quad j \in N \tag{2.1d}$$

$$y_j \in \{0, 1\}, \quad j \in N \tag{2.1e}$$

$$x_{ij} \in \{0, 1\}, \quad i \in M, j \in N. \tag{2.1f}$$

The objective function (2.1a) minimizes the facilities installation cost c_j and the supply costs c_{ij} between facility j and costumer i. The constraint set (2.1b) with the variable scope (2.1e) and (2.1f) imposes that only installed facilities can supply the costumers, and each costumer must be serviced by one facility. The constraint set (2.1c) imposes that all costumers must be supplied. The total demand cannot exceed the capacity b_j of a facility j in the constraint set (2.1d). Finally, variable y_j is set to 1 if facility j is installed and 0 otherwise, and variable x_{ij} is set to 1 if costumer i is supplied by facility j. The SSCFLP is an NP-hard problem, but becomes harder to solve than other NP-hard Facility Location Problem because the single-source constraints (KLOSE; DREXL, 2005).

Most of proposed methods to solve the SSCFLP use the Lagrangian relaxation as base for the solution method. The approaches consist in relaxing the constraint set (2.1b) or (2.1c), or both. For heuristic approaches, the work of Klincewicz (1986) relaxes the capacity constraint, which produces an uncapacitated facility location problem as the Lagrangian subproblem. This subproblem is solved with the dual ascent method of Erlenkotter (1978) and then heuristics are applied to obtain a primal feasible solution. The work of Hindi e PieŃkosz (1999) relaxes the single-source constraint in a Lagrangian manner and uses a heuristic in a restricted neighborhood to obtain the solutions for the problem. In Chen e Ting (2008), the solutions for SSCFLP are obtained using Lagrangian heuristic and a multiple ant colony system, which consists of two ant colony systems to be solved: the first selects the facilities that will be opened, and the second define the assignment of these facilities for each customer.

Other heuristic approaches for the SSCFLP can be found in the literature. The work of Delmaire, Díaz e Fenández (1999) presents a Tabu search, a reactive GRASP and two hybrid heuristics that combines Tabu search and reactive GRASP. The results were compared with the CPLEX and showed the efficiency of the proposed approaches, especially in the hybrid heuristics. More recently, Guastaroba e Speranza (2014) extended the Kernel Search heuristic framework to general binary integer linear programming problems and applied it to the SSCFLP. The idea of Kernel search consists of defining a set of promising variables (*kernel*) based on the relaxation of the problem that will be fixed for the next iteration. Also, is defined a set of no-promising variables called *bucket*, and at each iteration a variable in the *bucket* set is passed to *kernel* set, modifying the solution. The algorithm was tested in medium and large instances, obtaining the optimal solution in several instances. In Ho (2015) a simple Iterated Tabu search is proposed, which explores two neighborhoods and uses a penalization function for infeasible solutions. The algorithm was able to find optimal solutions in several instances, and when optimality was not found, it can obtain high-quality solutions.

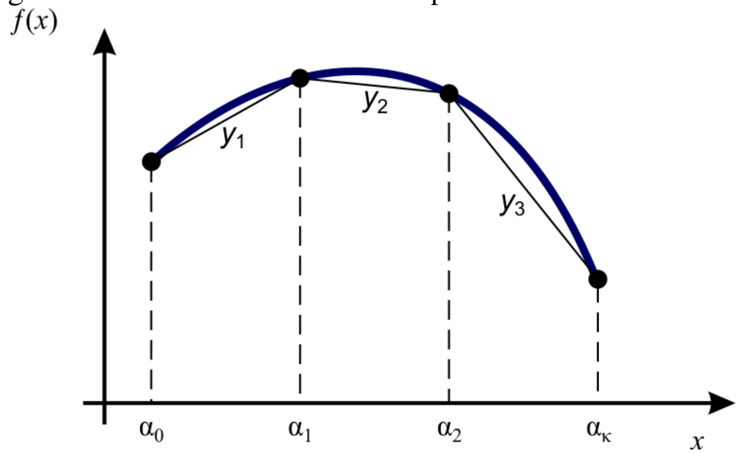
Exact approaches can be found in Holmberg, Rönnqvist e Yuan (1999), which propose a method based on a Lagrangian Dual Heuristic to compute lower bounds, plus a primal heuristic to generate upper bounds and a Branch-and-Bound algorithm that combines the dual and primal methods. Díaz e Fernández (2002) proposes a Column Generation algorithm that uses Lagrangian relaxation and incorporates it into a two-level Branch-and-Price framework. The first level selects facilities to be open and the second allocates the customers to the facilities. A Cut-and-Solve algorithm is proposed in Yang, Chu e Chen (2012), where a sparse problem with a small solution space is solved with a MIP solver, and cutting planes are applied to a relaxed dense problem to find tight lower bounds.

2.3 Piecewise-linear formulation

For solving optimization problems with nonlinear separable functions, a typical approach is the piecewise-linear reformulation of the functions, whereby a linear function and a set of constraints can be used to approximate the original function. Many techniques can be found in the literature, and some of them make the problem simpler by enabling it to be solved as a mixed integer linear problem (MILP). We present in this section two approaches for the piecewise-linear formulations that were applied in the CSP.

Consider Figure 2.2 and let $f(x)$ be a nonlinear and separable function, i.e., it can be written as a sum of functions that just depends on one variable. Also, let 3 the number of discretization points κ named from $\alpha_0, \ldots, \alpha_\kappa$, and $y_1, \ldots, y_\kappa$ be the number of segments in this discretization.

Figure 2.2 – Nonlinear function with piecewise-linear formulation



Source: from the author (2016).

The first approach was proposed by Beale e Tomlin (1970) and consists of defining a set of variables named as Special Ordered Set of type II (SOS2) for a convex combination. The SOS2 defines a ordered set of positive variables (ordered by a index), where at most two variables of the same set can be nonzero and they must be adjacent, *i.e.* just variables x_i and $x_{i+1}, i = 0, \ldots, \kappa - 1$. Considering a set of discretization points $K = \{k : k = 0, \ldots, \kappa\}$ and associating a variable λ_k for point α_k for each $k \in K$, we formulate the piecewise-linear model as follow:

$$
\begin{aligned}
x &= \sum_{k \in K} \lambda_k \alpha_k; && (2.2a)\\
f(x) &\approx \sum_{k \in K} \lambda_k f(\alpha_k); && (2.2b)\\
\text{s.t.} \quad & \sum_{k \in K} \lambda_k = 1; && (2.2c)\\
& \lambda_k \geq 0, \quad k \in K; && (2.2d)\\
& \{\lambda_k\}_{k \in K} \text{ is SOS2} && (2.2e)
\end{aligned}
$$

The constraint (2.2c) defines that only one segment $y_k : k = 1, \ldots, \kappa$ can be selected and by the SOS2 rules only the weights λ_k and λ_{k+1} will be positive for some $k = 0, \ldots, \kappa - 1$. When only one λ_k is positive, $x = \alpha_k$ and $f(x) = f(\alpha_k)$. Some solvers actually can implement this special structure for a set of variables.

The second approach for piecewise-linear formulations is known as Disaggregated Convex Combination (DCC) whose the properties are analyzed in (SHERALI, 2001). It is a pedagogically and simple modification of the model presented by (PADBERG, 2000). In the DCC, we define $K = \{k : k = 1, \ldots, \kappa\}$ and assign weights λ_k^L and λ_k^R for $k \in K$ to each endpoint of the segments $y_k \in K$. The elements of the nonlinear function can be replaced by:

$$
\begin{aligned}
x &= \sum_{k \in K} (\alpha_{k-1} \lambda_k^L + \alpha_k \lambda_k^R), && (2.3a)\\
f(x) &\approx \sum_{k \in K} [f(\alpha_{k-1}) \lambda_k^L + f(\alpha_k) \lambda_k^R], && (2.3b)\\
\text{s.t.} \quad & \lambda_k^L + \lambda_k^R = y_k, \quad k \in K, && (2.3c)\\
& \sum_{k \in K} y_k = 1, && (2.3d)\\
& \lambda_k^L, \lambda_k^R, y_k \geq 0, \quad k \in K, && (2.3e)\\
& y_k \in \{0, 1\}, \quad k \in K. && (2.3f)
\end{aligned}
$$

The model defines that at most one endpoint λ_k^L and λ_k^R can be positive and they must be of the same y_k segment in the constraint set (2.3c). The constraint set (2.3d) defines that just one y_k segment will be selected to approximate the nonlinear function.

2.4 Branch-and-Price

Branch-and-price is one of the most powerful techniques for solving huge integer and mixed integer problems. It embeds column generation into a branch-and-bound algorithm. We first give an overview of how column generation and branch-and-price works, and next we present some works in the literature that provide the basis for the implementation of these techniques.

CG algorithm was introduced in Gilmore e Gomory (1961) for solving the Cutting Stock Problem. In CG, we use the Dantzig-Wolfe decomposition (DANTZIG; WOLFE, 1960) to reformulate the compact formulation of the problem in a master problem (MP) such that the number of constraints is small, and the number of the variables (called columns) is large, possibly exponential in relation to size of the original problem. Since the size of problems that can be solved by MP directly is limited, the CG starts with a small initial set of these columns, and the MP is called Restricted Master Problem (RMP). In each iteration, the relaxation of RMP is solved and one or more sub-problems receive it dual values. The sub-problems are solved to find new columns that can improve the objective function of the RMP. CG ends when no more columns that improve the RMP objective can be found.

When the CG procedure ends, and the solution is integer, it is optimal for the original problem. Otherwise, it provides a lower bound (when considering a minimization problem) that is at least tighter or equal to the lower bound of relaxation of the compact model. To find the optimal integer solution we use the branch-and-price algorithm, which is a branch-and-bound framework, wherein at each node of enumeration tree a column generation procedure is solved.

B&P algorithm was first introduced by Desrosiers, Soumis e Desrochers (1984) for solving the vehicle routing problem with time windows. As the nodes are solved, they provide new bounds that can be used to prune the enumeration tree, which can helps for finding and proving the optimal solution. This method runs until all relaxed variables have an integer value. In the branch-and-bound, the branching rules are made on one variable. However in the B&P the branching rules must be on a set of variables, otherwise the algorithm may produce an unbalanced tree. Savelsbergh (1997) and Barnhart et al. (1998) give insights into the rules, also discussing the methodology and details for the CG and B&P implementation. Savelsbergh (1997) applies the B&P for the Generalized Assignment Problem, presenting comparatives for different pricing and proposed branching strategies. Barnhart et al. (1998) gives implementation examples also for the Generalized Assignment Problem and Crew Scheduling Problem.

Desaulniers, Desrosiers e Solomon (2005) presents in the first chapter a good introduction to the column generation method, a theoretical background such as the Dantzig-Wolfe decomposition, acceleration strategies and the use of Lagrangian relaxation with CG. The remaining chapters present an application of the branch-and-price for diverse problems such as vehicle routing, ship scheduling, machine scheduling, and job shop scheduling.

Lübbecke e Desrosiers (2005) present a survey of Column Generation, which includes the theoretical background for the decomposition principles of linear programming and convexification and discretization approaches for integer programming. In the algorithmic part, the authors explain the restricted master problem and the pricing subproblem, discussing implementation details for best performance.

3 THE COMPRESSOR SCHEDULING PROBLEM

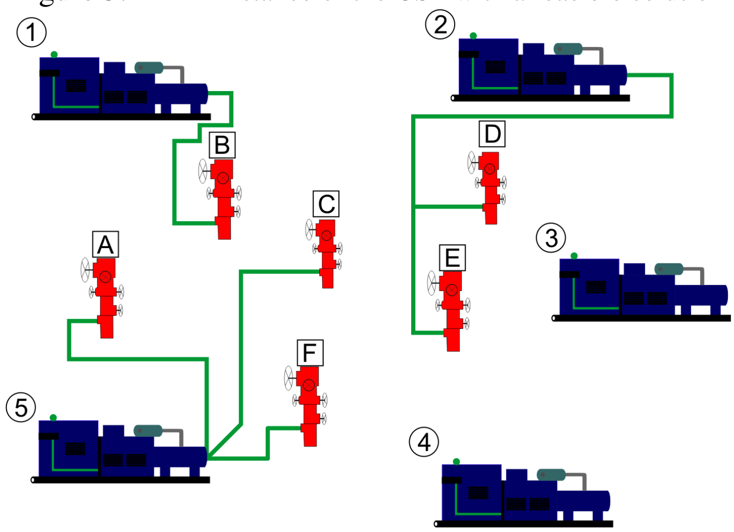
In this chapter, we present the definition, formulations and characteristics of the Compressor Scheduling Problem, along with the inherited structure of the Single-Source Capacitated Facility Location Problem, such as the sets and constraints. Moreover, we present two piecewise-linear formulations for the problem and some characteristics that we need to detail for the implementation of the proposed solution method.

3.1 Problem Definition

The CSP consists of defining among a set of gas-lift compressors, which will be installed/activated and determine which wells are serviced by each compressor such that the demand of gas rate and gas pressure of all wells are supplied. The objective is to minimize the installation(activation) and servicing(maintenance) costs, in addition to the compressors operating costs.

The compressors have a lower and upper gas rate production that must be respected, and a pressure function that varies according to the gas rate. Also, an energy loss cost is associated with each compressor. Moreover, there are different pressure drops in the pipeline which connect compressors to wells. We need to assure that the pressure provided by compressor is great than the pressure demand of well that it services, plus the pipeline pressure drops. Figure 3.1 illustrates a small instance and a feasible CSP solution.

Figure 3.1 – An instance of the CSP with a feasible solution



Source: from the author (2015).

Table 3.1 – Notation used in the CSP formulations.

Symbol	Definition
Sets	
$j \in N$	Set of compressors.
$i \in M$	Set of wells.
$j \in N_i$	Subset of compressors that can supply well i.
$i \in M_j$	Subset of wells that can be supplied by compressor j.
Parameters	
c_j	Installation/activation cost of compressor j.
d_j	Energy loss cost of compressor j.
$q_j^{c,min}$	Minimum output gas rate of compressor j.
$q_j^{c,max}$	Maximum output gas rate of compressor j.
$\alpha_{l,j}, l \in \{0,\ldots,4\}$	Parameters of discharge pressure of compressor j.
q_i^w	Gas rate demand of well i.
p_i^w	Gas pressure demand of well i.
c_{ij}	Cost of supply or maintenance between compressor j and well i.
l_{ij}	Pressure loss in the pipeline between compressor j and well i.
$q_j^{c,max,i}$	Maximum output gas rate q_j^c of compressor j at which the discharge pressure is sufficiently high to supply well i, *i.e.* $\max\{q_j^c : q_j^{c,min} \leq q_j^c \leq q_j^{c,max}, \ p_j^c(q_j^c) + l_{ij} \geq p_i^w\}$
Variables	
$y_j \in \{0,1\}$	Indicates whether the compressor j is installed.
$x_{ij} \in \{0,1\}$	Indicates whether the well i is supplied by compressor j.
$q_j^c \in \mathbb{R}_+$	Gas rate output of compressor j.
Functions	
$p_j^c(q_j^c)$	Discharge pressure output of compressor j, $p_j^c(q_j^c) = \alpha_{0,j} + \alpha_{1,j} q_j^c + \alpha_{2,j}(q_j^c)^2 + \alpha_{3,j}(q_j^c)^3 + \alpha_{4,j}\ln(1+q_j^c)$.
$h_j^c(q_j^c)$	Operating cost function of compressor j, $h_j^c(q_j^c) = d_j q_j^c p_j^c(q_j^c)$

Source: from the author (2016).

In Figure 3.1 there is a set of five compressors numbered from 1 to 5, and a set of six wells identified by the letters A to F. A feasible solution defines that compressors 1, 2 and 5 should be installed/activated. The well B is supplied by compressor 1, wells D and E are supplied by compressor 2, and wells A, C, and F are supplied by compressor 5. Note that, as explained earlier, each well must be supplied by only one compressor. With the cited characteristics the CSP can be considered an SSCFLP with two capacity constraints.

3.2 Formulations

The formulations presented from now use the notation presented from Table 3.1.

The CSP can be formulated as a Mixed-integer Nonlinear Problem (MINLP) as follow:

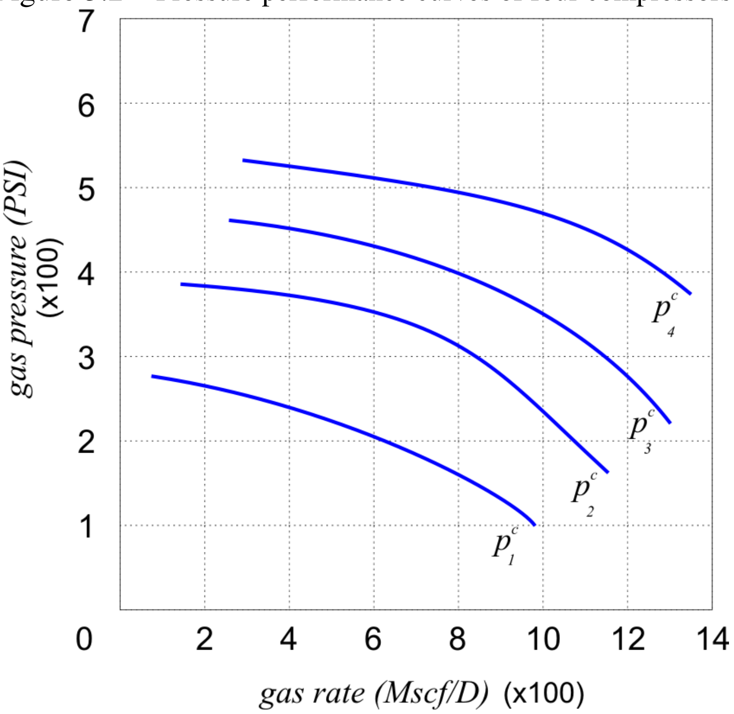
$$
\begin{aligned}
\min \quad P \;=\; & \sum_{j\in N} c_j y_j + \sum_{i\in M}\sum_{j\in N_i} c_{ij}x_{ij} + \sum_{j\in N} h_j^c(q_j^c) && \text{(3.1a)}\\
\text{s.t.} \quad : \; & x_{ij} \le y_j, \quad i \in M,\ j \in N_i && \text{(3.1b)}\\
& \sum_{j\in N_i} x_{ij} = 1, \quad i \in M && \text{(3.1c)}\\
& q_j^c \le q_j^{c,max,i} x_{ij} + q_j^{c,max}(y_j - x_{ij}), \quad j \in N, i \in M_j && \text{(3.1d)}\\
& q_j^c \ge q_j^{c,min} y_j, \quad j \in N && \text{(3.1e)}\\
& \sum_{i\in M_j} q_i^w x_{ij} \le q_j^c, \quad j \in N && \text{(3.1f)}\\
& q_j^c \ge 0, \quad j \in N && \text{(3.1g)}\\
& y_j \in \{0,1\}, \quad j \in N && \text{(3.1h)}\\
& x_{ij} \in \{0,1\}, \quad i \in M, j \in N_i. && \text{(3.1i)}
\end{aligned}
$$

The objective function minimizes the compressors installation/activation and operation costs and the supply/maintenance costs between compressors and wells. In the operating cost function, $p_j^c(q_j^c)$ is a decreasing function on q_j^c, *i.e.*, the compressor output pressure decreases as the outlet flow increases. Constraints set (3.1b) determines that a well is supplied only by an installed compressor. Constraints set (3.1c) imposes that all wells must be supplied. Constraints set (3.1d) impose the upper limit of the gas rate q_j^c. Note that if $x_{ij} = 1$ just the first term in the RHS of the inequality is activated, i.e. the output gas rate q_j^c must be less or equal to the $q_j^{c,max,i}$. Otherwise, the second term is activated, which imposes that the compressor j does not exceed its maximum output gas rate $q_j^{c,max}$. The constraints set (3.1e) imposes the lower limits of the compressor gas rate output. Constraints set (3.1f) imposes that the gas rate capacity of the compressor must be respected. Note that if $q_j^c > \sum_{i\in M_j} q_i^w x_{ij}$, we consider that the surplus gas is exported without cost increasing.

Note that if we consider only the two first terms in the objective (3.1a) and the constraint set (3.1b), (3.1c), (3.1f - replacing q_j^c by a parameter), (3.1h) and (3.1i), we have the same formulation of the SSCFLP. Thus, the CSP is an NP-hard problem as the SSCFLP and the proof can be found in Camponogara, Nazari e Meneses (2012).

The constraint set (3.1d) is a combination of linear inequalities proposed in Camponogara, Nazari e Meneses (2012) that replaces the constraint $p_j^c(q_j^c) \ge p_i^w + l_{ij}, j \in N, i \in M_j$ introduced in Camponogara, Castro e Plucenio (2007). This modification improves the model to be more efficient because the only nonlinear term in the Model (3.1) is the compressor op-

Figure 3.2 – Pressure performance curves of four compressors.



Source: from the author (2016).

erating costs in the objective function. Nevertheless, because of the non-convex objective, and as explained in the Section (2.3), the model (3.1) becomes very difficult to solve. A piecewise-linear formulation can be applied to handle this nonlinearity, as discussed in the next section.

3.2.1 Piecewise-linear formulation

The pressure performance function $p_j^c(q_j^c) = \alpha_{0,j} + \alpha_{1,j}q_j^c + \alpha_{2,j}(q_j^c)^2 + \alpha_{3,j}(q_j^c)^3 + \alpha_{4,j}\ln(1 + q_j^c)$, or simply p_j^c of a compressor j is concave and nonlinear as illustrated in Figure (3.2). From this, the following assumptions should be considered (CAMPONOGARA; PLUCENIO, 2008):

(i) p_j^c is concave, twice continuously differentiable function in the interval $q_j^c \in [0, q_{max}^c]$;

(ii) p_j^c decreases as q_j^c varies from $q_j^{c,min}$ to $q_j^{c,max}$;

(iii) $p_j^c \geq 0$ for all $q_j^c \in [0, q_j^{c,max}]$;

(iv) $q_i^w \leq q_j^{c,max}$ and $p_j^c(\max\{q_i^w, q_j^{c,min}\}) \geq p_i^w + l_{i,j}, \quad \forall i \in M_j$.

According to Camponogara e Plucenio (2008), the assumption (i) identify the typical shape of the output pressure and gas rate, and ensures the convexity of the feasible solutions space if p_j^c is concave between the interval $[q_j^{c,min}, q_j^{c,max}]$. The assumption (ii) shows the relation between pressure and rate: the rate increases as the pressure decreases. Assumption (iii) expresses that the pressure is nonnegative and assumption (iv) makes the sets M_j and N_i consistent.

We can reformulate the CSP through the piecewise-linear formulation, converting it to a mixed integer linear problem (MILP), which allows the use of techniques and algorithms off-the-shelf to solving MILPs. The piecewise-linear reformulation for the CSP was applied by Camponogara, Castro e Plucenio (2007), Camponogara e Plucenio (2008), Camponogara et al. (2011), Camponogara, Nazari e Meneses (2012) with the disaggregated convex combination. Moreover, Camponogara, Nazari e Meneses (2012) proposes the piecewise-linear formulation with SOS2 variables. In this work, are presented both cited models with the piecewise-linear formulation.

We assume that $Q_j = \{(q_j^{c,0}, h_j^{c,0}), \ldots, (q_j^{c,\kappa(j)}, h_j^{c,\kappa(j)})\}$ are the points of the compressor j with the output gas rate q_j^c and operating cost h_j^c, where:

(i) $\kappa(j)$ is the number of points in the piecewise-linear function;

(ii) $q_j^{c,k-1} < q_j^{c,k}$ for all $k \in K(j) = \{1, \ldots, \kappa(j)\}$;

(iii) $q_j^{c,0} = q_j^{c,min}$ and $q_j^{c,\kappa(j)} = q_j^{c,max}$ to maintain the output gas rate in a feasible range;

(iv) $h_j^{c,k} = d_j q_j^{c,k} p_j^c(q_j^{c,k})$ to be consistent with the piecewise-linear formulation.

From these assumptions and considering $K(j) = \{k : k = 1, \ldots, \kappa(j)\}$, the piecewise linear formulation for the CSP with the DCC model can be formalized as follow:

$$\min \quad \widetilde{P} = \sum_{j \in N} c_j y_j + \sum_{i \in M} \sum_{j \in N_i} c_{ij} x_{ij} + \sum_{j \in N} \widetilde{h}_j^{ue}(q_j^c) \tag{3.2a}$$

$$\text{s.t.} \quad \text{Equations (3.1b) to (3.1d) , (3.1f) to (3.1i)} \tag{3.2b}$$

$$q_j^c = \sum_{k \in K(j)} (q_j^{c,k-1} \lambda_j^{k,L} + q_j^{c,k} \lambda_j^{k,R}), \quad j \in N \tag{3.2c}$$

$$\sum_{k \in K(j)} z_j^k = y_j, \quad j \in N \tag{3.2d}$$

$$\lambda_j^{k,L} + \lambda_j^{k,R} = z_j^k, \quad j \in N, k \in K(j) \tag{3.2e}$$

$$\lambda_j^{k,L}, \lambda_j^{k,R} \geq 0, \quad j \in N, k \in K(j) \tag{3.2f}$$

$$z_j^k \in \{0, 1\}, \quad j \in N, k \in K(j), \tag{3.2g}$$

In the objective function (3.2a), $\widetilde{h}_j^{ue}(q_j^c) = \sum_{k \in K(j)} (h_j^{c,k-1} \lambda_j^{k,L} + h_j^{c,k} \lambda_j^{k,R})$ is an underestimator for each $j \in N$. The variables $\lambda_j^{k,L}$ and $\lambda_j^{k,R}$ are the weights of the left and right points of the segment from $q_j^{c,k-1}$ to $q_j^{c,k}$ which is represented by the variable z_j^k. Constraints set (3.2d) imposes that just one activated compressor j can have a selected segment to define q_j^c and $\widetilde{h}_j^{ue}(q_j^c)$.

Considering $K(j) = \{k : k = 0, \ldots, \kappa(j)\}$ the piecewise-linear model with the SOS2 variables can be formulated as follow:

$$\begin{aligned}
\text{Min} \quad & \widehat{P} = \sum_{j \in N} c_j y_j + \sum_{i \in M} \sum_{j \in N_i} c_{ij} x_{ij} + \sum_{j \in N} \widehat{h}_j^{ue}(q_j^c) && \text{(3.3a)} \\
\text{S.t.} \quad & \text{Equations (3.1b) to (3.1d) , (3.1f) to (3.1i)} && \text{(3.3b)} \\
& q_j^c = \sum_{k \in K(j)} q_j^{c,k} \lambda_j^k, \quad j \in N && \text{(3.3c)} \\
& \sum_{k \in K(j)} \lambda_j^k = y_j, \quad j \in N && \text{(3.3d)} \\
& \lambda_j^k \geq 0, \quad j \in N, k \in K(j) && \text{(3.3e)} \\
& \{\lambda_j^k\}_{k \in K(j)} \text{ is SOS2}, \quad j \in N && \text{(3.3f)}
\end{aligned}$$

In the objective function (3.3a), $\widehat{h}_j^{ue}(q_j^c) = \sum_{k \in K(j)} h_j^{c,k} \lambda_j^k$ is also an under-estimator for the original compressor operation cost. In the constraint set (3.3d) the variables $\lambda_j^k, k \in K(j)$ can be nonzero only if the compressor j is activated.

Models (3.2) and (3.3) are equivalent and both provide a lower-bound for the model (3.1) due the under-estimators $\widetilde{h}_j^{ue}(q_j^c)$ and $\widehat{h}_j^{ue}(q_j^c)$. The approximation depends on the number $\kappa(j)$ of the points in the piecewise-linear function, *i.e.* as $\kappa(j)$ increases, more accurate the approximations $\widetilde{P}$ and $\widehat{P}$ become with respect to P. On the other hand, $\kappa(j)$ influence in the number of variables and constraints in the formulations with piecewise-linear models. Model (3.2) has $\Theta(3|N|\kappa(j))$ more variables and $\Theta(2|N| + \kappa(j)|N|)$ more constraints than Model (3.1). Model (3.3) has $\Theta(\kappa|N|)$ more variables and $\Theta(2|N|)$ more constraints than Model (3.1). Thus, there is important to define a minimum number $\kappa(j)$ for each $j \in N$ that has a good approximation, and consequently consumes less processing time as possible.

4 COLUMN GENERATION ALGORITHM

Solving models (3.2) and (3.3) directly with a MIP solver can take a long time and may not provide an optimal solution, especially when considering large and more difficult instances of the CSP, as we can see in the experiments reported in Chapter 6. In this chapter, we present a column generation approach for the CSP, where the problems $\widetilde{P}$ and $\widehat{P}$ are reformulated into a master problem with subproblems based on the Dantzig-Wolfe decomposition.

4.1 Master problem

A column in CSP is an assignment S_j of compressor j to wells that is represented as a pair (y, x), where $y = (y_j : j \in N)$ and $x = (x_{ij} : i \in M_j, j \in N_i)$ are vectors associated with the decision variables. $A_j^\ddagger \subseteq 2^{M_j}$ is the set that represents all feasible assignments of a compressor $j \in N$. An assignment $S_j \in A_j^\ddagger$ if $\sum_{i \in S_j} q_i^w \leq q_j^{c,max}$ and $\max\{\sum_{i \in S_j} q_i^w, q_j^{c,min}\} \leq \min\{q_j^{c,max,i} : i \in S_j\}$. The gas rate of column S_j is $q_{S_j}^c = \max\{\sum_{i \in S_j} q_i^w, q_j^{c,min}\}$ and the assignment cost is defined by $c_{S_j} = c_j + \sum_{i \in S_j} c_{ij} + h_j^c(q_{S_j}^c)$. Thus, the master problem (MP) can be formulated as follow:

$$
\begin{aligned}
P_{MP} = \min \quad & \sum_{j \in N} \sum_{S_j \in A_j^\ddagger} c_{S_j} \lambda_{S_j} && \text{(4.1a)} \\
\text{s.t.} : \quad & \sum_{j \in N} \sum_{S_j \in A_j^\ddagger} \delta_{S_j}^i \lambda_{S_j} = 1, \quad i \in M && \text{(4.1b)} \\
& \sum_{S_j \in A_j^\ddagger} \lambda_{S_j} \leq 1, \quad j \in N && \text{(4.1c)} \\
& \lambda_{S_j} \in \{0, 1\}, \quad j \in N, S_j \in A_j^\ddagger && \text{(4.1d)}
\end{aligned}
$$

The objective function (4.1a) minimizes the cost of the assignments such that all wells must be supplied in the constraints set (4.1b), and at most only one assignment per compressor can be selected in the constraint set (4.1c). The coefficient $\delta_{S_j}^i$ is equal to 1 if well i is supplied by compressor j in assignment S_j, and 0 otherwise. The binary variable λ_{S_j} takes the value 1 if column S_j is selected, and 0 otherwise.

Remark 1. *Problems P and P_{MP} have the same optimal solution with hypothesis that $q_j^c = \sum_{i \in M_j} q_i^w x_{ij}$ for all $j \in N : y_j = 1$. Both problems are equivalent if the inequality in constraint set (3.1f) can be replaced by an equality.*

Proof. Suppose a compressor j is activated in an optimal solution for P, and $q_j^c = \sum_{i \in M_j} q_i^w x_{ij}$ in P_{MP} and $q_j^c > \sum_{i \in M_j} q_i^w x_{ij}$ in P. As the constraint set (3.1f) requires that the compressor gas rate must be at least equal to the demand supplied to wells, and problems P and MP consider the value q_j^c in its objective function, the only reason for q_j^c to be greater than the demand supplied to the wells is that the operating cost $h_j^c(q_j^c)$ is lesser for P than MP. However, the solution obtained by reducing q_j^c to $\sum_{i \in M_j} q_i^w x_{ij}$ is feasible in view of assumption (i) and with higher cost, contradicts the hypothesis. □

4.1.1 The Restricted Master Problem

The straightforward resolution of the Model (4.1) with all $S_j \in A_j^{\ddagger}, j \in N$, is not efficient due the possibly exponential number of columns on the number of wells in the set $A_j^{\ddagger}, j \in N$. Furthermore, the optimal solution will select at most $|N|$ columns, and all others will have the associated variable set to 0 due the definition of the constraint set (4.1c) and (4.1d). To handle these issues, the column generation procedure starts with an initial set of columns in the master problem and adds new columns iteratively. This leads to Restricted Master Problem (RMP) defined as follow:

$$
\begin{aligned}
P_{RMP} = \min \quad & \sum_{j \in N} \sum_{S_j \in A_j^{\dagger}} c_{S_j} \lambda_{S_j} && \text{(4.2a)} \\
\text{s.t.} : \quad & \sum_{j \in N} \sum_{S_j \in A_j^{\dagger}} \delta_{S_j}^i \lambda_{S_j} = 1, \quad i \in M && \text{(4.2b)} \\
& \sum_{S_j \in A_j^{\dagger}} \lambda_{S_j} \leq 1, \quad j \in N && \text{(4.2c)} \\
& \lambda_{S_j} \in \{0, 1\}, \quad j \in N, S_j \in A_j^{\dagger}, && \text{(4.2d)}
\end{aligned}
$$

where $A_j^{\dagger} \subseteq A_j^{\ddagger}$ is any subset of feasible columns for compressor j. The initial subset of columns is defined by Algorithm 1.

Algorithm 1 creates for each compressor a minimum number of columns S_j such that all clients in M_j are supplied at least once. For example, if the first column S_j of a compressor j supplies the wells $i = \{1, 2\} \in M_j$, necessarily the second column will supply the wells from $i = \{3, \ldots, |M_j|\}$, or until the gas $q_{S_j}^c$ will not be sufficient to supply the well demand and so on. At line (2) the algorithm creates the first column S_j for the compressor j with no wells assigned to it. The algorithm then enters a loop for each well $i \in M_j$ (line 3) and verifies if the

Algorithm 1 Initial columns for the RMP.

1: **for** each $j \in N$ **do**
2:     Create a column S_j for compressor j with $\delta^i_{S_j} = 0, \forall i \in M_j$ and $q^c_{S_j} = 0$;
3:     **for** each $i \in M_j$ **do**
4:         **if** ($q^c_{S_j} + q^w_i \leq q_j^{c,max,i}$ and $q^c_{S_j} + q^w_i \leq q_j^{c,max,l}, \forall l \in S_j\{i\}$) **then**
5:             $q^c_{S_j} = q^c_{S_j} + q^w_i$;
6:             $\delta^i_{S_j} = 1$;
7:         **else**
8:             Create a new column S_j for compressor j with $\delta^i_{S_j} = 0, \forall i \in M_j$;
9:             $q^c_{S_j} = q^w_i$;
10:             $\delta^i_{S_j} = 1$;
11:         **end if**
12:     **end for**
13:     Create an artificial $\lambda^\star_{S_j}$, with $c_{S_j} = (\sum_{j \in N} c_j)^2$ and $\delta^i_{S_j} = 1, \forall i \in M$;
14: **end for**

current column S_j has sufficient gas rate to supply well i (line 4). If it has, then well i will be supplied in this column at lines 5 and 6, where the algorithm updates the variables $q^c_{S_j}$ and $\delta^i_{S_j}$ of current column S_j. Otherwise, a new column S_j is initialized for this compressor (lines 8 to 10).

The initial RMP must have a feasible relaxation solution to ensure that proper information is passed to the subproblems (BARNHART et al., 1998). To ensure it, the algorithm inserts an artificial variable $\lambda^\star_{S_j}$ for each $j \in N$ in the line (13). If at least one $\lambda^\star_{S_j} > 0$ at the end of the CG procedure, it means that the real columns are not sufficient to compose a solution for the CSP, then it is infeasible. Although the initial columns without the artificial variables can produce a feasible relaxed solution, we use it to ensure the feasibility of relaxation of restricted master problem in the Branch-and-Price nodes.

4.2 The pricing sub-problem

With the initial columns, we solve the linear relaxation of P_{RMP}. The optimal relaxed solution provides dual variables $\pi_i, i \in M$, associated with the constraint set (4.2b), and dual variables $\mu_j, j \in N$, associated with the constraint set (4.2c). This information is used by the pricing subproblem to find the columns with minimum reduced cost for each $j \in N$ as follows:

$$SP_j : \bar{c}_j = \min_{S_j \in A^\ddagger_j} c_{S_j} - \sum_{i \in S_j} \pi_i - \mu_j \tag{4.3}$$

Notice that $\bar{c}_j$ is the least reduced cost for compressor j. If $\min\{\bar{c}_j : j \in N\} \geq 0$, then the linear relaxation of P_{RMP} is optimal for the relaxation of P_{MP}. Otherwise, new columns with negative reduced cost that can decrease the value of P_{RMP} must be added.

However, SP_j can be used for analyzing only the columns that are in the RMP. As we do not have all columns in the RMP, then we need to look for new columns solving the pricing subproblem indirectly, reformulating it as a MILP based on the piecewise-linear formulation Models (3.2) or (3.3). Considering $K(j) = \{k : k = 1, \ldots, \kappa(j)\}$, the pricing subproblem that approximates SP_j based on the piecewise-linear formulation model (3.2), can be formulated as follow:

$$\widetilde{SP}_j : \widetilde{c}_j = \min \quad c_j - \mu_j + \sum_{i \in M_j} (c_{ij} - \pi_i) x_i + \widetilde{h}_j^{ue}(q_j^c) \tag{4.4a}$$

$$\text{s.t.} \quad : \quad q_j^c = \sum_{k \in K(j)} (q_j^{c,k-1} \lambda^{k,L} + q_j^{c,k} \lambda^{k,R}) \tag{4.4b}$$

$$q_j^c \leq q_j^{c,max,i} x_i + q_j^{c,max}(1 - x_i), \quad i \in M_j \tag{4.4c}$$

$$\sum_{i \in M_j} q_i^w x_i \leq q_j^c \tag{4.4d}$$

$$\sum_{k \in K(j)} z_j^k = 1 \tag{4.4e}$$

$$\lambda_j^{k,L} + \lambda_j^{k,R} = z_j^k, \quad k \in K(j) \tag{4.4f}$$

$$\lambda_j^{k,L}, \lambda_j^{k,R} \geq 0, \quad k \in K(j) \tag{4.4g}$$

$$z_j^k \in \{0, 1\}, \quad k \in K(j) \tag{4.4h}$$

$$x_i \in \{0, 1\}, \quad i \in M_j, \tag{4.4i}$$

where $\widetilde{c}_j$ is the reduced cost of $\widetilde{SP}_j$. The new column is associated with the variables x. Note that the under-estimator $\widetilde{h}_j^{ue}(q_j^c) = \sum_{k \in K(j)} (h_j^{c,k-1} \lambda_j^{k,L} + h_j^{c,k} \lambda_j^{k,R})$ is the same used from Model (3.2).

Similarly, considering $K(j) = \{k : k = 0, \ldots, \kappa(j)\}$, the pricing sub-problem that approximates SP_j based on the piecewise-linear formulation model (3.3), can be formulated as follow:

$$\widehat{SP}_j : \widehat{c}_j = \min \quad c_j - \mu_j + \sum_{i \in M_j} (c_{ij} - \pi_i) x_i + \widehat{h}_j^{ue}(q_j^c) \tag{4.5a}$$

$$\text{s.t.} \quad : \quad q_j^c = \sum_{k \in K(j)} q_j^{c,k} \lambda^k \tag{4.5b}$$

$$q_j^c \leq q_j^{c,max,i} x_i + q_j^{c,max}(1 - x_i), \quad i \in M_j \tag{4.5c}$$

$$\sum_{i \in M_j} q_i^w x_i \leq q_j^c \tag{4.5d}$$

$$\sum_{k \in K(j)} \lambda_j^k = 1 \tag{4.5e}$$

$$\lambda_j^k \geq 0, \quad k \in K(j) \tag{4.5f}$$

$$\{\lambda_j^k\}_{k \in K(j)} \text{ is SOS2} \tag{4.5g}$$

In the objective function, the under-estimator $\widehat{h}_j^{ue}(q_j^c) = \sum_{k \in K(j)} h_j^{c,k} \lambda_j^k$ is the same from Model (3.3).

As the Models (4.4) and (4.5) are equivalent, the following observations apply for both, but we consider just the notation of the first Model.

Remark 2. *The subproblem $\widetilde{SP}_j$ is consistent with the formulation of MP only if inequality (4.4d) is replaced by an equality. The solution of $\widetilde{SP}_j$ may lead to higher gas production than the amount consumed by the wells supplied by compressor j.*

Proof. Similarly to the Remark 1 and considering assumptions (i) to (iii), the solution of $\widetilde{SP}_j$ may lead to a $q_j^c > \sum_{i \in M_j} q_i^w x_i$ which implies in a compressor operations cost $h_j^c(q_j^c)$ lower than the operating cost with $q_j^c = \sum_{i \in M_j} q_i^w x_i$. □

Remark 3. *The use of the under-estimator $\widetilde{h}_j^{ue}(q_j^c)$ is relevant to calculate the reduced costs $\widetilde{c}_j$. As $\widetilde{h}_j^{ue}(q_j^c) \leq h_j^c(q_j^c)$, if $\widetilde{c}_j \not< 0$, there is no column with negative reduced cost that can improve the solution for P_{RMP}.*

Remark 4. *If $\widetilde{c}_j < 0$, we need to recalculate the effective reduced cost $\overline{c}_j$, which is obtained by using function $h_j^c(q_j^c)$ instead of the under-estimator $h_j^{ue}(q_j^c)$. If $\overline{c}_j < 0$, the column can be added to P_{RMP}. If $\overline{c}_j \not< 0$, there are two possibilities. The first considers that $\widetilde{SP}_j$ is a good approximation for SP_j, then there is no column of compressor j that can be added to P_{RMP}. The second considers that $\widetilde{SP}_j$ is not a good approximation for SP_j. Then, the under-estimator $h_j^{ue}(q_j^c)$ should be refined, increasing the number of points $\kappa(j)$.*

Remark 5. *Problems $\widetilde{P}$ and $\widehat{P}$ are equivalent to problem P_{RMP} if we consider the use of the under-estimator $\widetilde{h}_j^{ue}(q_j^c)$ in the column cost c_{S_j} instead of the compressor operation cost $h_j^c(q_j^c)$.*

Proof. The first and second term of the column cost $c_{S_j} = c_j + \sum_{i \in S_j} c_{ij} + h_j^c(q_{S_j}^c)$ are equivalent to the first and second terms of the objective function in models (3.2) and (3.3). The unique difference is $h_j^c(q_j^c)$ in the column cost and $\widetilde{h}_j^{ue}(q_j^c)$ in the cited models. If we use $\widetilde{h}_j^{ue}(q_j^c)$ for both models, they become equivalent. □

4.3 Column Generation procedure

An iteration of column generation consists in solving the linear relaxation of P_{RMP}, then solving $\widetilde{SP}_j$ for each $j \in N$. Considering Remark 4 and assuming a good approximation of $\widetilde{SP}_j$ to SP_j, Algorithm 2 describes the column generation procedure for the CSP.

Algorithm 2 Column Generation procedure.

1: **procedure** CG_ITER( )
2:     **repeat**
3:         Solve the linear relaxation of P_{RMP};
4:         LB = optimal relaxed objective value of P_{RMP};
5:         (π, μ) = optimal dual solution of the linear relaxation of P_{RMP};
6:         **for** each $j \in N$ **do**
7:             Solve $\widetilde{SP}_j$ with (π, μ);
8:             **if** $(\widetilde{c}_j < 0)$ **then**
9:                 Calculate the corresponding SP_j with q_j^c and x generated in the line 7;
10:                 **if** $(\overline{c}_j < 0)$ **then**
11:                     RMP $\leftarrow$ RMP $\cup$ {column generated by $\widetilde{SP}_j$};
12:                 **end if**
13:             **end if**
14:         **end for**
15:     **until** $(\overline{c}_j : j \in N) > 0$, for all j.
16:     return LB;
17: **end procedure**

In Algorithm 2, we first solve the linear relaxation of Model (4.2) in line 3. Next, n pricing subproblems are solved (line 7) considering the dual values obtained in line 5. If the reduced cost of the $\widetilde{SP}_j$ is smaller than 0 we calculate the effective $\overline{c}_j$ (line 9). If $\overline{c}_j$ is negative, we insert the column generated by $\widetilde{SP}_j$ in the RMP. If one or more pricing have negative reduced cost, the linear relaxation of RMP is solved again, now with the newly inserted column(s). This procedure runs until all effective reduced costs return a value greater or equal to 0, which means that the relaxation of MP is optimal, and its objective value is returned in the line 16.

5 BRANCH-AND-PRICE

The optimal solution of the restricted master problem relaxation is optimal for the original problem if all $\lambda_{S_j} = \{0, 1\}, S_j \in A_j^\dagger,\ j \in N$. Otherwise, in the case of a relaxed solution, we find the optimal integer solution with a Branch-and-Price procedure, which divides the original search space into two subproblems (or nodes) with an additional constraint. As the RMP may not contain all optimal columns, we solve a column generation procedure in each node. In this chapter, we describe in detail the B&P proposed for the CSP.

5.1 Branching strategy

Differently from Branch-and-Bound, the simple branching on only one variable λ_{S_j} may produce an unbalanced tree (BARNHART et al., 1998) for the B&P. It is desirable to define a branching rule in a set of variables that can divide the problem in a symmetric way. For this purpose, we define two branching rules that were applied depending on the characteristic of the relaxed solution in the current node. These branch rules are based on the original variables of problem P, given in Model (3.1).

The first rule is a branching on the compressors, or simply **branch (A)**, that forbids the activation of a compressor j in one branch and requires that this same compressor must be activated in the other branch.

Considering the optimal solution of P_{RMP}, we define $t_j = \sum_{S_j \in A_j^\dagger} \lambda_{S_j}$ for each $j \in N$. If at least one t_j has a fractional value, we define the compressors with maximum and minimum value of t_j as follow:

$$j^{max} = \arg\max_{j \in N} \{t_j : 0 < t_j < 1\} \tag{5.1a}$$

$$j^{min} = \arg\min_{j \in N} \{t_j : 0 < t_j < 1\} \tag{5.1b}$$

With the compressors obtained in (5.1a) and (5.1b), we define j^a as the compressor j closer to integrality as follow:

$$j^a = \arg \max_{j \in \{j^{max}, j^{min}\}} \{1 - t_{j^{min}}, t_{j^{max}}\} \tag{5.2}$$

Finally, we branching in the left (L) and right (R) nodes as follow:

$$\text{L}: \sum_{S_{j^a} \in A^{\dagger}_{j^a}} \lambda_{S_j} = 1 \qquad \text{R}: \sum_{S_{j^a} \in A^{\dagger}_{j^a}} \lambda_{S_j} = 0 \tag{5.3a}$$

Equation (5.3a) defines that in left branch the compressor j^a must be activated in the solution, and in the right branch the compressor j^a must be deactivated. If $t_{j^a} > 0.5$ the left branch is explored first, otherwise the right branch first is explored first.

Note that we just need to change the constraint set (4.2c) to apply the branch (A) for P_{RMP}. For the left branch, we replace the inequality to an equality in the j^a-*th* constraint. For the right branch, we made the same procedure as the left branch, and change the right-hand side (RHS) from 1 to 0 in the j^a-*th* constraint. Thus, the restricted master problem with the **left** branch (A) applied for the compressor j^a becomes as follow:

$$\text{(A) left}: P_{RMP} = \text{Eq.(4.2a)} \tag{5.4a}$$

$$\text{s.t.} : \text{Eq. (4.2b) and (4.2d)} \tag{5.4b}$$

$$\sum_{S_j \in A^{\dagger}_j} \lambda_{S_j} \leq 1, j \in N \backslash \{j^a\} \tag{5.4c}$$

$$\sum_{S_{j^a} \in A^{\dagger}_{j^a}} \lambda_{j^a S_{j^a}} = 1 \tag{5.4d}$$

The restricted master problem with the **right** branch (A) applied for the compressor j^a becomes as follow:

$$\text{(A) right}: P_{RMP} = \text{Eq.(4.2a)} \tag{5.5a}$$

$$\text{s.t.} : \text{Eq. (4.2b) and (4.2d)} \tag{5.5b}$$

$$\sum_{S_j \in A^{\dagger}_j} \lambda_{S_j} \leq 1, j \in N \backslash \{j^a\} \tag{5.5c}$$

$$\sum_{S_{j^a} \in A^{\dagger}_{j^a}} \lambda_{j^a S_{j^a}} = 0 \tag{5.5d}$$

As the right branch forbids the compressor j^a to be activated, we do not need to execute the $\widetilde{SP}_{j^a}$ in the column generation procedure.

Remark 6. *If the relaxed solution of P_{RMP} is fractional and all $t_j \in \{0, 1\}, j \in N$, we cannot apply the branch (A) because no j^{min} and j^{max} will be found, but there may be columns with fractional λ_{S_j} values.*

Remark 6 is important to define a second branching rule, the branch on wells, or simply **branch (B)**. Considering again the optimal relaxed solution of P_{RMP}, we define the columns with greater and lesser λ_{S_j} value as follow:

$$S_j^{max} = \arg \max_{S_j \in A_j^\dagger, j \in N} \left\{ \lambda_{S_j} : 0 < \lambda_{S_j} < 1 \right\} \tag{5.6a}$$

$$S_j^{min} = \arg \min_{S_j \in A_j^\dagger, j \in N} \left\{ \lambda_{S_j} : 0 < \lambda_{S_j} < 1 \right\} \tag{5.6b}$$

After, we define S_j^b as the column with the closest value to integrality as follow:

$$S_j^b = \arg \max_{S_j \in \{S_j^{min}, S_j^{max}\}} \{1 - \lambda_{S_j^{min}}, \lambda_{S_j^{max}}\} \tag{5.7}$$

Let j^b the compressor $j \in N$ which column S_j^b is associated. Also, let i^b be a well that is supplied by column S_j^b, *i.e.*, $\delta_{S_{j^b}^b}^{i^b} = 1$ which is not restricted to any compressor. The branch (B) rules are defined for the P_{RMP} as follow:

$$\text{L} : \sum_{S_j \in A_{j^b}^\dagger : \delta_{S_{j^b}}^{i^b} = 1} \lambda_{S_{j^b}} = 1, \qquad \text{R} : \sum_{S_j \in A_{j^b}^\dagger : \delta_{S_{j^b}}^{i^b} = 1} \lambda_{S_{j^b}} = 0 \tag{5.8a}$$

In Equation (5.8a), we define that well i^b is restricted to be supplied by compressor j^b in the left branch, and forbid compressor j^b to supply well i^b in the right branch. If $\lambda_{S_j^b} > 0.5$, the left branch is explored first, otherwise the right branch first is explored first.

For branch (B), we need to create new constraint sets for Model (4.2). Thus, we define another RMP that replaces P_{RMP} as follow:

$$P_{RMP^1} = \min \sum_{j \in N} \sum_{S_j \in A_j^\dagger} c_{S_j} \lambda_{S_j} \tag{5.9a}$$

$$\text{s.t.} : \sum_{j \in N} \sum_{S_j \in A_j^\dagger} \delta_{S_j}^i \lambda_{S_j} = 1, \quad i \in M \tag{5.9b}$$

$$\sum_{S_j \in A_j^\dagger} \lambda_{S_j} \leq 1, \quad j \in N \tag{5.9c}$$

$$\sum_{S_j \in A_j^\dagger : \delta_{S_j}^i = 1} \lambda_{S_j} \leq 1, \quad j \in N, i \in M_j \tag{5.9d}$$

$$\lambda_{S_j} \in \{0, 1\}, \quad j \in N, S_j \in A_j^\dagger, \tag{5.9e}$$

The constraints set (5.9d) is used for branch (B). Note that in the first node, these constraints are redundant with respect to constraint set (5.9c). As the branch-and-price evolves, the constraints can be replaced by an equality, and the RHS can assume value 0 in the right branch.

Adding new constraints to the RMP implies in new dual values for the optimal relaxation of P_{RMP^1} that must be considered in the pricing subproblems. We re-define the pricing for the compressor j as follow:

$$\widetilde{SP}_j^1 : \widetilde{c}_j^1 = \text{Min} \quad : \quad c_j - \mu_j + \sum_{i \in M_j} (c_{ij} - \pi_i - \psi_{ij}) x_i + \widetilde{h}_j^{ue}(q_j^c) \tag{5.10a}$$

$$\text{S.t.} \quad : \quad \text{Equations (4.4b) to (4.4i)} \tag{5.10b}$$

where ψ_{ij} is the dual variable associated with the ij-th constraint (5.9d) of P_{RMP^1}.

The branch (B) rule implies in changing the bounds of the variables x in the pricing subproblems to produce feasible columns for the current node. If compressor j must supply well i, variable x_i is fixed to 1 in the $\widetilde{SP}_j$, and fixed to 0 for all $\widetilde{SP}_l : l \in N \backslash j$. When compressor j is forbidden to supply well i, we just set x_i to zero in $\widetilde{SP}_j$.

Remark 7. *It is possible that all wells $i \in M_{j^b} : \delta^i_{S^b_j} = 1$ are already restricted to be supplied by compressor j^b due previous branch rules. Thus, defining a well i^b and compressor j^b from column S^b_j leads to the following cases for the branch (B):*

1. *The left branch will impose that a well i^b must be supplied by compressor j^b. However, this rule was applied in a previous branch and is redundant. Then the feasible solution space will be the same and does not change the solution of P_{RMP^1}.*

2. *The right branch will impose that compressor j^b not supply well i^b. However, a previously branch rule imposed the opposite for the same j^b and i^b. Then, the new branch rule will replace the previously branch rule, which no make sense in a branch-and-price enumeration tree.*

Then, there is no possibility to apply branch (B) for compressor j^b and any well $i \in M_{j^b}$ supplied by the column S^b_j.

To handle Remark 7, we just re-define S^b_j as the column with seccond closest value λ_{S_j} to integrality (S^{min}_j or S^{max}_j). Assuming that $S^b_j = S^{min}_j$ but cannot compose a branching rule, and defining $R = \{i \in M : \delta^i_{S^{min}_j} = 1\}$ as the set of wells supplied in the column S^{min}_j, then S^{max}_j will have at least one $i \in M : \delta^i_{S^{max}_j} = 1$ that is not restricted yet, in two cases as follows:

1. S^{max}_j and S^{min}_j are columns of the same compressor $j \in N$: Assuming that there is no repeated columns for the same compressor, the column S^{max}_j supplies at least one $i \notin R$ that is not restricted for this compressor, otherwise $S^{min}_j = S^{max}_j$ or $\lambda_{S^{min}_j} = 1$;

2. S_j^{max} and S_j^{min} are columns of different compressor $j \in N$: This is implicit because each $i \in R$ is not supplied in S_j^{max} because the branch rules, and is not possible that exists a column that does not meet any well $i \in M$.

5.2 Branch-and-Price procedure

The branch-and-price procedure is outlined in Algorithm 3.

Algorithm 3 Branch-and-Price procedure

1: **procedure** BRANCH-AND-PRICE(*P, UB*)
2: x = *Solve the relaxation of problem P using Column Generation*;
3: **if** *(x is infeasible or $\phi(x)$ > UB)* **then**
4: **return** $\emptyset$;
5: **end if**
6: **if** *(x is integral)* **then**
7: **if** *($\phi(x)$ < UB)* **then**
8: *UB* = $\phi(x)$; ▷ Update the upper bound
9: **end if**
10: **return** $\emptyset$;
11: **end if**
12: y = *Solve the integer restricted master problem with CPLEX*;
13: **if** *($\phi(y)$ < UB)* **then**
14: *UB* = $\phi(y)$; ▷ Update the upper bound
15: **end if**
16: *Define the $branch\_rule$ as branch (A) or branch (B);*
17: **if** *($guide\_value$ > 0.5)* **then**
18: $B = \{L, R\}$;
19: **else**
20: $B = \{R, L\}$;
21: **end if**
22: **for** *all* $b \in B$ **do**
23: *Add* $branch\_rule(B[b])$ *to P*;
24: BRANCH-AND-PRICE(*P, UB*) ;
25: *Remove* $branch\_rule(B[b])$ *from P*;
26: **end for**
27: *Return the solution associated with the UB*;
28: **end procedure**

In Algorithm 3 the search tree is explored using a depth-first strategy to find feasible solutions to the CSP as fast as possible, using its values as the upper bound. The algorithm first execute the Column Generation procedure of Algorithm 2 in line 2. If the solution x is infeasible or its objective value $\phi(x)$ is greater than the upper bound UB, the algorithm cuts the node per infeasibility (line 4). If solution x is integer, then UB is updated if its possible (line 8) and cut the node per integrality in line 10. If the algorithm does not return $\emptyset$ after line 10, the problem need branching. In line 12 the restricted master problem (5.9) is solved as an integer problem to produce a feasible solution and verify if it can be used as a better UB (lines 13 to 15). In line 16 the algorithm analyzes solution x and define which branch (A) or branch (B) will be applied as $branch\_rule$. $guide\_value = t_{j^a}$ if branch (A) is choosed, and $guide\_value = \lambda_{S_j^b}$ if branch (B) is choosed. According to the $guide\_value$ the algorithm defines the order to explore the branches (L = Left, R = Right) in lines 17 to 21. Then, in line 23 the corresponding $branch\_rule$ to problem P is added and the Branch-and-Price is solved recursively in line 24. After exploring the sub-tree, the $branch\_rule$ is removed in the line 25. The algorithm ends after no more calls from the branch-and-price procedure can be made, which means that all nodes are explored. At this point, the algorithm returns the optimal solution in line 27.

In Algorithm 3 any column added in any node of the enumeration tree remains in the restricted master problem for the other sub-trees, *i.e.* a column is never removed from the RMP. Also, the artificial columns $\lambda^*_{S_j}, j \in N$ are not considered in the branching rules.

5.3 Speedups

In Lasdon (1970) a procedure to calculate a valid lower bound is defined for the restricted master problem. Considering Z^v as the current objective value of relaxation of Model (5.9) at iteration v of Algorithm (2), and the reduced cost $\overline{c_j}^v$ of each $j \in N$ pricing subproblem at the same v iteration. We can define the RMP lower bound LB^v for the CSP as follow:

$$LB^v = \min_{j \in N} \overline{c_j}^v |N| + Z^v; \tag{5.11}$$

As the effort to calculate LB^v is minimal, we can calculate it in each iteration of the column generation. As the number of iterations v is incremented, the value LB^v tends to be tighter. Then if the $LB^v > UB$ at some node of the branch-and-price tree, this node can be cut off before the end of the column generation procedure, which can save execution time.

6 COMPUTATIONAL RESULTS

This chapter presents the computational results obtained by solving the CSP. First we compare the lower bounds of the Column Generation with the linear relaxation of Model (3.2). Next we compare the exact solution obtained by our Branch-and-Price and the CPLEX Solver. Both are tested with the two piecewise-linear formulations presented to compare their efficiency.

6.1 Data sets

The instance tests consist of three sets: Set 1 is composed of four instances of the CSP provided in Nazari (2011) and three instances provided by Camponogara, Nazari e Meneses (2012) identified by index from 1 to 7. Sets 2 and 3 are instances of the SSCFLP, available in Holmberg, Rönnqvist e Yuan (1999) and Delmaire, Díaz e Fenández (1999), identified by index 8 to 19, and 20 to 27 respectively. These sets were extended to the CSP, using as reference the first instance of set 1. The piecewise-linear formulation uses $\kappa(j) = 10, j \in N$ for all instances. To transform a SSCFLP instance into a CSP instance, let l^c the smaller and g^c the higher gas rate values of all compressors $j \in N$ in the reference instance. Accordingly, let l^w the smallest and g^w the highest values of gas rate demands of all wells $i \in M$ in the reference instance.

The function $rand(x, y)$ represents a random value between x and y, where the seed value used corresponds to the index of facility/customer. The correspondence of each parameter is described in Table 6.1

The models were solved in a computer with AMD-FX-8150 processor, running at 3.6 GHz on a single core, with 32 GB RAM. The operational system was the Ubuntu 13.04, kernel 3.8.0-27, and the algorithms were implemented in C++ with the CPLEX API 12.5.0.

We first compare in Section 6.2 the lower-bounds provided by the column generation and the linear relaxation of Model (3.2). Next in the sections 6.3 and 6.4 we present the results for the CPLEX solver and the proposed B&P, respectively. For each analysis, we report results for each instance, running with time limit of 300 seconds, 1, 5 and 10 hours. First we present the comparative results of the piecewise-linear formulation with binary and SOS2 variables for CPLEX and the B&P approaches. Next, we compare the best results of each approach.

Table 6.1 – Transformation of SSCFLP instances to CSP instances.

Parameter name	SSCFLP	CSP
Facility & Compressor		
Installation cost:	c_j	c_j
Energy cost loss	-	$d_j = rand(1,7)$
Capacity	b_j	$q_j^{c,min} = b_j/rand([b_j/l^c],[b_j/g^c])$
Capacity	b_j	$q_j^{c,max} = b_j/rand(0.11, 0.44)$
Pressure parameter	-	$\alpha_{0,j} = q_j^{c,max} * rand(0.2887, 0.7983)$
Pressure parameter	-	$\alpha_{1,j} = rand(-1.215, -0.14)$
Pressure parameter	-	$\alpha_{2,j} = \alpha_{1,j} * rand(-0.12, -0.09)$
Pressure parameter	-	$\alpha_{3,j} = \alpha_{1,j} * rand(0.0061, 0.0138)$
Pressure parameter	-	$\alpha_{4,j} = \alpha_{1,j} * rand(-1.6258, -0.976)$
Client & Well		
Demand	d_i	$q_i^w = d_i/rand([d_i/l^w],[d_i/g^w])$
Demand	-	$p_i^w = rand(0.74, 5)$
Client/Well x Facility/Compressor		
Supply cost	c_{ij}	c_{ij}
Pipeline pressure drops	-	$l_{ij} = 0.1$

Source: from the author (2016).

6.2 Lower bounds results

Although the SOS2 is an implicit model of the piecewise-linear approximation and we need to remove these sets in the problem $\widehat{P}$ to obtain the linear relaxation, the relaxation of models (3.2) and (3.3) are equivalent, and produced the same lower bounds. Similarly, the lower bounds obtained by column generation using pricing subproblem models (4.4) or (4.5) are the same. Thus, in Table 6.2 we just present the lower bounds obtained by the relaxation of Model (3.2) in the column LB_{DCC}, and in the column LB_{CG} we present the lower bound obtained by column generation using the pricing subproblem $\widehat{SP}_j$. Column *I* represents the index of the instance, column *N* shows the number of compressors, and column *M* presents the number of wells. The column *BKV* represents the best-known value (BKV) found in our experiments, and is marked with a '*' when it is known to be optimal. For each lower bound approach, we present in column *Time(s)* the processing time to obtain the lower bound. In the *GAP(%)* column, we calculate the lower bound relative deviation $100(\frac{BKV-LB}{LB})$ from the best-known value, and LB represents the value of the lower bound found for each approach. The lines below the last instance of each set correspond to the average values obtained in the set, and the last line of the table presents the average values from the three sets.

For LB_{CG} we present in the columns *RMP(%)* and $SP(\%)$ the portion of the total time spend to solve the relaxation of the restricted master problem and the pricing subproblems,

Table 6.2 – CSP Lower bounds

I	*N*	*M*	*BKV*	LB_{DCC}		LB_{CG}				
				Time(s)	*GAP(%)*	*Time(s)*	*GAP(%)*	*RMP(%)*	$SP(\%)$	*#Cols*
1	5	6	*274.86	0.02	22.26	0.10	0.00	0.00	61.39	29
2	7	16	*428.90	0.02	37.52	0.56	4.98	0.00	86.83	97
3	8	18	*434.30	0.01	27.74	1.39	0.00	0.00	92.45	138
4	9	14	*360.98	0.01	34.24	0.44	2.19	0.00	83.56	90
5	14	20	*411.14	0.01	23.47	1.90	0.89	0.00	89.12	232
6	14	32	*585.55	0.01	16.15	6.62	0.98	0.08	94.81	397
7	31	64	*1,003.17	0.12	21.97	17.37	0.39	0.51	94.73	851
				0.03	26.19	4.06	**1.35**	0.08	86.13	262.00
8	20	50	*18,278.43	0.03	27.91	17.46	0.47	0.28	97.45	739
9	20	50	*15,581.52	0.03	28.52	14.83	0.41	0.32	97.07	725
10	20	50	*19,505.43	0.03	25.65	18.36	0.46	0.28	97.50	726
11	20	50	*23,305.43	0.04	24.12	19.99	0.42	0.27	97.65	749
12	20	50	*18,278.43	0.03	27.91	17.47	0.47	0.29	97.50	739
13	20	50	*15,581.52	0.03	28.52	14.84	0.41	0.32	97.07	725
14	20	50	*19,505.43	0.03	25.65	18.38	0.46	0.29	97.47	726
15	20	50	*23,305.43	0.04	24.12	20.00	0.42	0.27	97.65	749
16	20	50	*18,278.43	0.03	27.91	17.46	0.47	0.28	97.45	739
17	20	50	*15,581.52	0.03	28.52	14.85	0.41	0.32	97.07	725
18	20	50	*19,505.43	0.03	25.65	18.38	0.46	0.29	97.55	726
19	20	50	*23,305.43	0.04	24.12	19.99	0.42	0.27	97.64	749
				0.03	26.55	17.67	**0.44**	0.29	97.42	734.75
20	30	60	15,240.21	0.09	4.31	21.98	0.67	0.71	94.62	1,442
21	30	60	18,297.27	0.09	18.66	72.53	1.59	0.18	98.63	1,241
22	30	60	57,526.87	0.12	18.59	86.21	2.07	0.13	98.87	1,233
23	30	60	51,954.48	0.11	21.01	88.13	2.47	0.15	98.89	1,287
24	30	60	63,895.96	0.12	19.11	99.23	1.61	0.14	98.93	1,268
25	30	60	64,722.34	0.11	17.52	85.32	0.95	0.15	98.83	1,250
26	30	60	91,149.77	0.12	16.68	85.01	1.22	0.16	98.86	1,298
27	30	60	212,516.16	0.13	26.27	93.55	3.58	0.13	98.99	1,252
				0.11	17.77	78.99	**1.77**	0.22	98.32	1,283.88
Avg				**0.06**	23.50	33.57	**1.19**	0.20	93.96	760.21

Source: from the author (2016).

respectively. In the column *#Cols* is presented the total number of generated assignments S_j for all $j \in N$, including the initial columns, the artificial columns, and the columns generated by solving $\widehat{SP}_j, j \in N$.

In Table 6.2 we mark in bold the technique that obtained the best average time computation and the best average GAP. We observe that average GAP of the column generation is less than 2% from the best-known solution. Although LB_{CG} requires more computational time, it is almost 20 fold tighter than the lower bounds of obtained in LB_{DCC}. Furthermore, the column generation found the optimal solution in the instances 1 and 3. We can observe that the pricing subproblems consume almost all processing time as they are solved as a MILP, while the RMP consume an insignificant portion of the processing time (it is solved as linear problem).

6.3 CPLEX results

In Table 6.3 we present the results obtained by CPLEX for solving Model (3.2), named *DCC*, and Model (3.3), named *SOS2*. The column *Time(s)* presents the CPU time in seconds spent to solve each instance with a limit of 36,000 seconds (10 hours). The column *GAP(%)* presents the percentage deviation $(\frac{S-BKV}{BKV})100$, where S represents the objective value from the best feasible solution found by the solver and BKV represents the best-known value. The 'NFS' marked in the column *GAP(%)* means that no feasible solution was found by the solver within the time limit. Similarly to Table 6.2, after the last instance of each set, we present the average values of the set, and in the last line, the average values from all instance sets are presented.

In Table 6.3 we mark in bold the best results between the two models. When an optimal solution is found we mark in bold the formulation that was solved in a shorter time. Otherwise, we mark the model that found the smallest deviation. When no bold mark appears on the line, we consider that the results are equivalent. Analyzing Table 6.3 we observe that the model *SOS2* found the optimal solution in shorter time than *DCC* model only for the instance 1 . However, this time is insignificant. In other four cases (instances 7, 9, 13 and 17) the results with the specified time limit are equivalent. For the remaining 22 instances, the model *DCC* had better results than model *SOS2*, both in *Time(s)* and *GAP(%)*.

Table 6.4 presents the average results for solving the CSP with the solver CPLEX. We show the results with four different time limits as described in the column *Time Limit*. The column *Instance Set* indexes the average results of the three instance sets. Column *#Optimal* present the notation a/b, where a is the number of instances that were solved until optimality, and b is the total number of instances in the set. Column *Time to opt(s)* shows the average time in seconds to reach these optimal values. With same notation of column *#Optimal*, the column *#Feasible Solutions* indicates the number of instances that integer feasible solutions were found in the specified time limit. The average percentage deviation *GAP(%)* was calculated in the same way that in Table 6.3. But in this case, the *GAP(%)* considers only the instances where the optimality was not proved, and an integer feasible solution was found.

Table 6.3 – CSP results with the CPLEX Solver.

I	*N*	*M*	*BKV*	*DCC*		*SOS2*	
				Time(s)	*GAP (%)*	*Time(s)*	*GAP (%)*
1	5	6	*274.86	0.06	0.00	**0.03**	0.00
2	7	16	*428.90	**1.73**	0.00	2.29	0.00
3	8	18	*434.30	**1.12**	0.00	2.00	0.00
4	9	14	*360.98	**1.12**	0.00	2.10	0.00
5	14	20	*411.14	**4.09**	0.00	13.06	0.00
6	14	32	*585.55	**79.02**	0.00	1,014.21	0.00
7	31	64	*1,003.17	36,000.02	0.00	36,000.50	0.00
				5,155.31	0.00	5,290.60	0.00
8	20	50	*18,278.43	**3,447.65**	0.00	36,000.03	0.01
9	20	50	*15,581.52	36,000.07	0.00	36,000.01	0.00
10	20	50	*19,505.43	36,000.01	**0.01**	36,000.04	0.02
11	20	50	*23,305.43	**7,245.18**	0.00	36,001.97	0.00
12	20	50	*18,278.43	**3,352.56**	0.00	36,000.59	0.01
13	20	50	*15,581.52	36,000.02	0.00	36,000.03	0.00
14	20	50	*19,505.43	36,000.01	**0.00**	36,000.03	0.02
15	20	50	*23,305.43	**7,323.04**	0.00	36,002.85	0.00
16	20	50	*18,278.43	**3,224.70**	0.00	36,000.59	0.01
17	20	50	*15,581.52	36,000.02	0.00	36,000.03	0.00
18	20	50	*19,505.43	36,000.03	**0.00**	36,000.03	0.01
19	20	50	*23,305.43	**7,277.82**	0.00	36,000.86	0.00
				20,655.92	0.00	36,000.59	0.01
20	30	60	15,240.21	36,079.66	**0.06**	36,000.85	0.06
21	30	60	18,297.27	36,000.64	**0.00**	36,003.40	0.00
21	30	60	57,526.87	36,001.17	**0.03**	36,000.66	NFS
23	30	60	51,954.48	36,000.82	**0.02**	36,001.82	NFS
24	30	60	63,895.96	36,001.03	**0.02**	36,000.79	0.07
25	30	60	64,722.34	36,000.17	**0.09**	36,001.66	0.09
26	30	60	91,149.77	36,000.90	**0.05**	36,064.94	NFS
27	30	60	212,516.16	36,000.19	**0.01**	36,000.18	0.03
				36,010.57	**0.03**	36,009.29	0.05
Avg				20,607.27	**0.01**	25,766.82	0.02

Source: from the author (2016).

In Table 6.4, the bold mark represents which of the two models had the best results when they can be compared directly. For example, if *#Optimal* are equal for *Binary* and *SOS2*, and represents the same instances, there is a draw and we compare the results marking in bold the best *Time to opt(s)*. Similarly, if the number of feasible solutions found for each approach is equal, we compare the average GAP of both and mark the best with bold. The '-' represents that no solution was found in the columns *#Optimal* and *#Feasible Solutions*. In the columns *Time to opt(s)* and *GAP(%)* the same mark '-' means that the corresponding value cannot be calculated, *i.e.* if no optimal solution was found, the *Time to opt(s)* will be '-', and if no feasible solutions were found the *GAP(%)* will be '-'.

Table 6.4 – CSP average results with CPLEX Solver.

Time Limit	*Instance Set*	*#Optimal*		*Time to opt (s)*		*#Feasible Solutions*		*GAP (%)*	
		DCC	*SOS2*	*DCC*	*SOS2*	*DCC*	*SOS2*	*DCC*	*SOS2*
300 s	1	**6/7**	5/7	14.65	3.89	7/7	7/7	0.00	0.01
	2	-	-	-	-	**12/12**	-	0.02	-
	3	-	-	-	-	**8/8**	5/8	0.10	0.12
1 h	1	6/7	6/7	**14.65**	172.28	7/7	7/7	**0.00**	0.01
	2	**3/12**	-	3351.74	-	**12/12**	9/12	0.01	0.01
	3	-	-	-	-	**8/8**	5/8	0.04	0.05
5 h	1	6/7	6/7	**14.65**	172.28	7/7	7/7	0.00	0.00
	2	**6/12**	-	5334.53	-	12/12	12/12	**0.00**	0.01
	3	-	-	-	-	**8/8**	5/8	0.04	0.05
10 h	1	6/7	6/7	**14.65**	172.28	7/7	7/7	0.00	0.00
	2	**6/12**	-	5334.53	-	12/12	12/12	**0.00**	0.01
	3	-	-	-	-	**8/8**	5/8	0.03	0.05

Source: from the author (2016).

As we can see in Table 6.4, the *DCC* Model (3.2) found feasible integer solutions for all instances with a time limit of 300 seconds, while no feasible solution was found for 3 instances of set 3 by the Model *SOS2* (3.3), even with a time limit of 10 hours. The time to reach the optimal solution can be compared directly only for the instance Set 1, where the formulation with binaries variables obtained better results. For the instance Set 2, the optimality was reached for 6 out of the 12 instances by the model *DCC* in the time limit of 5 hours. With a time limit of 10 hours, the optimal value was found in 11 instances of this set, but the optionality was not proved. Similarly, although the model *SOS2* cannot prove the optimality for any instance of set 2 with a time limit of 10 hours, the optimal value was found in three instances. For the instance set 3, no optimal solutions was found in both models and the *DCC* model obtained best integer solutions.

As more optimal solutions have been found by Model (3.2) in the specified running times than Model (3.3), and considering that the integer feasible solutions are better, we consider the first model for comparing the results to the branch-and-price.

6.4 Branch-and-price results

In Table 6.5 we compare the results for the CSP obtained by our Branch-and-Price with the pricing sub-problem models (4.4) and (4.5), represented in the tables as columns BP_{DCC} and BP_{SOS2}, respectively. Also, we present in *RMP(%)* the portion of the total time spent to

Table 6.5 – CSP results with Branch-and-Price: Times and gap

I	*N*	*M*	*BKV*	BP_{DCC}				BP_{SOS2}			
				Time(s)	*GAP (%)*	*RMP(%)*	*SP(%)*	*Time(s)*	*GAP (%)*	*RMP(%)*	*SP(%)*
1	5	6	*274.86	0.34	0.00	0.00	84.30	**0.09**	0.00	0.00	60.47
2	7	16	*428.90	28.59	0.00	6.05	90.03	**9.12**	0.00	5.71	84.66
3	8	18	*434.30	2.97	0.00	0.00	95.04	**1.38**	0.00	0.00	92.17
4	9	14	*360.98	8.13	0.00	2.73	93.23	**2.76**	0.00	4.81	84.59
5	14	20	*411.14	28.38	0.00	6.47	89.21	**15.91**	0.00	10.09	82.22
6	14	32	*585.55	36,000.13	0.00	56.31	41.37	36,000.01	0.00	35.60	20.23
7	31	64	*1,003.17	36,001.12	0.00	71.74	27.34	36,001.13	0.00	72.00	27.03
				10,295.66	0.00	20.47	74.36	**10,290.06**	0.00	18.32	64.48
8	20	50	*18,278.43	179.79	0.00	3.55	95.14	**143.33**	0.00	4.64	93.83
9	20	50	*15,581.52	219.74	0.00	4.56	93.93	**111.85**	0.00	5.23	92.85
10	20	50	*19,505.43	**169.37**	0.00	4.05	94.70	203.71	0.00	6.42	91.93
11	20	50	*23,305.43	539.53	0.00	7.86	90.72	**407.77**	0.00	8.00	90.47
12	20	50	*18,278.43	165.96	0.00	3.55	95.05	**143.43**	0.00	4.63	93.82
13	20	50	*15,581.52	189.22	0.00	4.50	93.80	**111.90**	0.00	5.23	92.85
14	20	50	*19,505.43	**145.29**	0.00	4.06	94.52	203.90	0.00	6.42	91.93
15	20	50	*23,305.43	538.78	0.00	7.87	90.73	**407.87**	0.00	7.99	90.48
16	20	50	*18,278.43	187.16	0.00	3.51	95.22	**143.67**	0.00	4.64	93.82
17	20	50	*15,581.52	222.57	0.00	4.53	94.00	**112.01**	0.00	5.23	92.83
18	20	50	*19,505.43	**178.85**	0.00	4.02	94.79	203.84	0.00	6.42	91.92
19	20	50	*23,305.43	518.03	0.00	7.76	90.75	**407.91**	0.00	8.00	90.47
				271.19	0.00	4.99	93.61	**216.77**	0.00	6.07	92.26
20	30	60	15,240.21	36,007.81	0.00	73.03	26.35	36,005.43	0.00	69.32	29.92
21	30	60	18,297.27	36,015.57	0.01	94.72	5.19	36,005.22	0.01	89.71	10.15
22	30	60	57,526.87	36,004.73	**0.00**	99.47	0.52	36,010.62	0.01	99.42	0.57
23	30	60	51,954.48	36,004.50	0.00	99.35	0.65	36,013.76	**0.00**	99.42	0.58
24	30	60	63,895.96	36,008.82	0.00	99.37	0.58	36,012.57	0.00	99.47	0.53
25	30	60	64,722.34	36,011.07	**0.00**	98.85	1.14	36,009.24	0.04	99.57	0.42
26	30	60	91,149.77	36,021.06	0.00	25.94	73.28	36,006.69	0.00	29.00	70.30
27	30	60	212,516.16	36,009.87	0.00	99.44	0.56	36,008.53	0.00	99.42	0.58
				36,010.43	**0.00**	86.27	13.53	36,009.01	0.01	85.67	14.13
Avg				15,525.76	0.00	37.24	60.50	**15,505.28**	0.00	36.68	56.96

Source: from the author (2016).

solve the Model (5.9), including the time for solving the RMP as an integer problem. Column *SP(%)* represents the portion of time spent to solve subproblem $\widetilde{SP}_j$ for BP_{DCC} and subproblem $\widehat{SP}_j$ for BP_{SOS2}. The others columns are already explained for the tables 6.2 and 6.3.

In Table 6.5 we mark in bold the best results in relation of *Time(s)* when optimality was reached, otherwise we mark the best results in the column *GAP(%)*. When a line has no bold mark, we cannot compare the approaches by these two columns.

Concerning the time to reach the optimal solution, the BP_{SOS2} approach needs less time in 14 out of the 17 cases than the BP_{DCC} considering the sets 1 and 2. A statical test of Wilcoxon signed-rank with a significance level of 0.05 obtain a p-value = 0.0075, which means that we accept the hypothesis that BP_{DCC} takes longer than the BP_{SOS2} to find the optimal solutions. The percentage deviation of both approaches is the same on seven of ten cases that

Table 6.6 – CSP results with Branch-and-Price: Number of nodes, cuts and columns.

I	*N*	*M*	BP_{DCC}			BP_{SOS2}		
			#Nodes	*#Cuts*	*#Cols*	*#Nodes*	*#Cuts*	*#Cols*
1	5	6	1	1	28	1	1	29
2	7	16	107	54	329	81	41	337
3	8	18	1	1	145	1	1	138
4	9	14	27	14	134	27	14	150
5	14	20	51	26	391	51	26	417
6	14	32	55,217	27,580	9,496	43,317	21,629	8,108
7	31	64	7,488	3,730	12,103	7,611	3,794	14,144
			8,984.57	4,486.57	3,232.29	7,298.43	3,643.71	3,331.86
8	20	50	63	32	977	59	30	996
9	20	50	95	48	987	55	28	939
10	20	50	51	26	989	95	48	1,116
11	20	50	233	117	1,491	203	102	1,428
12	20	50	63	32	977	59	30	996
13	20	50	95	48	987	55	28	939
14	20	50	51	26	989	95	48	1,116
15	20	50	233	117	1,491	203	102	1,428
16	20	50	63	32	977	59	30	996
17	20	50	95	48	987	55	28	939
18	20	50	51	26	989	95	48	1,116
19	20	50	233	117	1,491	203	102	1,428
			110.50	55.75	1,111.00	103.00	52.00	1,119.75
20	30	60	7,851	3,924	8,311	10,535	5,266	9,028
21	30	60	445	220	2,803	1,193	593	3,576
21	30	60	15	0	1,276	17	0	1,275
23	30	60	15	0	1,347	18	0	1,352
24	30	60	14	0	1,328	18	0	1,320
25	30	60	35	0	1,399	11	0	1,282
26	30	60	2,403	1,177	10,665	2,260	1,108	11,265
27	30	60	12	0	1,301	13	0	1,300
			1,348.75	665.13	3,553.75	1,758.13	870.88	3,799.75
Avg			3,481.27	1,735.82	2,632.35	3,053.18	1,522.20	2,750.45

Source: from the author (2016).

the optimal solution was not found. Also, the BP_{DCC} obtained two better integer solutions than the BP_{SOS2} approach.

For both B&P approaches, the portion of processing time spent to solve the RMP grows in relation to the results in Table 6.2, especially in the large instances, because the time for solving the integer restricted master problem is also recorded in the column *RMP(%)*.

In Table 6.6 we present other comparative results for BP_{DCC} and BP_{SOS2} approaches. The column *#Nodes* presents the number of explored nodes in the B&P tree, and column *#Cuts* shows the number of applied cuts in the B&P tree, *i.e.* cuts by infeasibility and optimality. The column *#Cols* presents the total number of columns at the end of branch-and-price.

From the results of Table 6.6, we can observe that the columns *#Nodes*, *#Cuts* and *#Cols* diverge between the approaches. Analyzing the logs, it happens because the Solver CPLEX considers values with a large number of decimal cases that can diverge after the 8th decimal digit for the same parameter. It can occur in a column cost c_{S_j}, and consequently, the relaxation of the master problem can produce symmetric solutions in each approach, which can affect the branch rule, and the branch-and-price tree can be explored in a different way.

Similarly to Table 6.4, Table 6.7 presents the average results obtained with the branch-and-price considering the pricing sub-problem Model 4.4 in the column BP_{DCC}, and the pricing sub-problem model 4.5 in the column BP_{SOS2}.

Table 6.7 – CSP average results with Branch-and-Price.

Time Limit	*Instance Set*	*#Optimal* BP_{DCC}	BP_{SOS2}	*Time to opt (s)* BP_{DCC}	BP_{SOS2}	*#Feasible solutions* BP_{DCC}	BP_{SOS2}	*GAP (%)* BP_{DCC}	BP_{SOS2}
	1	5/7	5/7	13.68	**5.85**	7/7	7/7	0.00	**0.00**
300 s	2	9/12	9/12	184.22	**153.07**	12/12	12/12	0.01	0.01
	3	-	-	-	-	8/8	8/8	**0.01**	0.02
	1	5/7	5/7	13.68	**5.85**	7/7	7/7	0.00	0.00
1 h	2	12/12	12/12	271.19	**216.76**	12/12	12/12	-	-
	3	-	-	-	-	8/8	8/8	**0.00**	0.01
	1	5/7	5/7	13.68	**5.85**	7/7	7/7	0.00	0.00
5 h	2	12/12	12/12	271.19	**216.76**	12/12	12/12	-	-
	3	-	-	-	-	8/8	8/8	**0.00**	0.01
	1	5/7	5/7	13.68	**5.85**	7/7	7/7	0.00	0.00
10 h	2	12/12	12/12	271.19	**216.76**	12/12	12/12	-	-
	3	-	-	-	-	8/8	8/8	**0.00**	0.01

Source: from the author (2016).

In Table 6.7 the number of optimal and integer feasible solutions is equal for both approaches in all time limits defined. Then we compare the models in a sense of time to reach the optimal solution, and the average GAP of the integer feasible solutions found. As earlier observed in Table 6.5, the average time limit to find the optimal solution was lesser for the BP_{SOS2} than BP_{DCC} approach.

About the average GAP for the integer feasible solutions, just with the time limit of 300 seconds the average results are better for the BP_{SOS2} in the instance set 1. When the runtime is extended, the BP_{DCC} have the same average GAP for the instance set 1 and can find better integral solutions than the BP_{SOS2} for the instance set 3, including four best-known values.

For the next section we elect the result of BP_{SOS2} to compare with the CPLEX solver.

6.5 Comparative Results

Table 6.8 presents a comparison between our Branch-and-Price and the CPLEX solver for solving the Compressor Scheduling Problem. As explained earlier, we compare the B&P that uses the pricing subproblem Model (4.5) with SOS2 variables and the DCC MILP Model (3.2) for the piecewise-linear formulation in the columns *B&P* and *CPLEX*, respectively. The results are compared in relation to time in seconds to obtain the optimal solution (column *Time(s)*) and the relative deviation of the integral solution found (column *GAP(%)*). We present only the results of the execution with the time limit of 10 hours.

Table 6.8 – CSP Results: Branch-and-Price vs CPLEX

I	*N*	*M*	*BKV*	*B&P*		*CPLEX*	
				Time(s)	*GAP (%)*	*Time(s)*	*GAP (%)*
1	5	6	*274.86	0.09	0.00	**0.06**	0.00
2	7	16	*428.90	9.12	0.00	**1.73**	0.00
3	8	18	*434.30	1.38	0.00	**1.12**	0.00
4	9	14	*360.98	2.76	0.00	**1.12**	0.00
5	14	20	*411.14	15.91	0.00	**4.09**	0.00
6	14	32	*585.55	36,000.01	0.00	**79.02**	0.00
7	31	64	*1,003.17	36,001.13	0.00	36,000.02	0.00
				10,290.06	0.00	**5,155.31**	0.00
8	20	50	*18,278.43	**143.33**	0.00	3,447.65	0.00
9	20	50	*15,581.52	**111.85**	0.00	36,000.07	0.00
10	20	50	*19,505.43	**203.71**	0.00	36,000.01	0.01
11	20	50	*23,305.43	**407.77**	0.00	7,245.18	0.00
12	20	50	*18,278.43	**143.43**	0.00	3,352.56	0.00
13	20	50	*15,581.52	**111.90**	0.00	36,000.02	0.00
14	20	50	*19,505.43	**203.90**	0.00	36,000.01	0.00
15	20	50	*23,305.43	**407.87**	0.00	7,323.04	0.00
16	20	50	*18,278.43	**143.67**	0.00	3,224.70	0.00
17	20	50	*15,581.52	**112.01**	0.00	36,000.02	0.00
18	20	50	*19,505.43	**203.84**	0.00	36,000.03	0.00
19	20	50	*23,305.43	**407.91**	0.00	7,277.82	0.00
				216.77	0.00	20,655.92	0.00
20	30	60	15,240.21	36,005.43	**0.00**	36,079.66	0.06
21	30	60	18,297.27	36,005.22	0.01	36,000.64	**0.00**
21	30	60	57,526.87	36,010.62	**0.01**	36,001.17	0.03
23	30	60	51,954.48	36,013.76	**0.00**	36,000.82	0.02
24	30	60	63,895.96	36,012.57	**0.00**	36,001.03	0.02
25	30	60	64,722.34	36,009.24	**0.04**	36,000.17	0.09
26	30	60	91,149.77	36,006.69	**0.00**	36,000.90	0.05
27	30	60	212,516.16	36,008.53	**0.00**	36,000.19	0.01
				36,009.01	**0.01**	36,010.57	0.03
	Avg			**15,505.28**	**0.00**	20,607.27	0.01

Source: from the author (2016).

From the results of Table 6.8, CPLEX outperforms our B&P in the first six instances in instance set 1. In particular in instance 6, the B&P cannot find the optimal solution in 10 hours of processing, while the CPLEX can find the solution in approximately 80 seconds. For instance 7 the obtained solution was not proved optimal for both approaches, and the integer solution value was the same.

Considering instance set 2, our B&P can find all optimal solutions in less than 10 minutes, while CPLEX found the optimal value just for six instances with a time greater than 55 minutes for the instances 8, 12 and 16, and a processing time greater than 2 hours for the other tree instances (11, 15, and 19). For the other six instances (9-10, 13-14, and 17-18), although CPLEX found the optimal value for five of them, the optimality cannot be proved since the solver cannot apply any cut in the branch-and-cut procedure. This issue should be investigated.

In the instance set 3, the results showed that the B&P found integer solutions sensibly better in general than the CPLEX, and just in one instance CPLEX found the best-known value. We extended the time limit to 24 hours for the set 3 and the instance 7. We observe that CPLEX can prove the optimal solution for the *BKV = 1,003.17* in instance 7 in approximately 20 hours while branch-and-price cannot prove the optimality, although it found the same objective value as the best integer. On the other hand, there were no improvement in any instance of the set 3 in both CPLEX and B&P. However, CPLEX stopped earlier than the stipulated time limit due the memory overflow in 6 instances.

Finally, the global average values of our Branch-and-Price outperformed CPLEX in time and average deviation, which is an important result.

7 CONCLUSION

Decisions problems in the oil production receive considerable attention in research areas as combinatorial optimization since these problems can impact on the costs of the oil production, and consequently in the profits of the petroleum industry. The Compressor Scheduling Problem consists in defining for a production field which gas-lift compressors will be installed or activated to supply a set of wells, minimizing the associated costs. The problem is an extension of the classical Single-Source Capacitated Facility Location Problem and can be classified as a mixed integer nonlinear problem that was converted into a mixed integer linear problem by the piecewise-linear formulation.

This work proposed a Branch-and-Price algorithm to solve the CSP. Two piecewise-linear formulations were tested in the pricing subproblem to handle the nonlinear function, showing that the Special Ordered Set of type two obtained better results than the disaggregated convex combination approach. Moreover, the lower bounds achieved by the column generation procedure were up to 20 times tighter than the linear relaxation of the compact model for some instances.

The computational experiments we tested for three instance sets, out of which two sets we generated based on instances of the SSCFLP. We compared the results of the B&P with the commercial solver CPLEX, and although the second obtained better results than our algorithm for the set 1, the first found an optimal solution faster for the set 2 and obtained better integer feasible solutions for the third set, which no optimal solution was found in the stipulated time limit. Another remark is that CPLEX consumes more memory than the B&P and can compromise the quality of solution in larger instances.

As future work we intend to improve the branch-and-price algorithm, using a two-phase approach for solving faster the pricing subproblem as suggested in Savelsbergh (1997), Barnhart et al. (1998), using a heuristic to find columns with negative reduced cost and solving the MILP problem just to prove that the RMP cannot be improved in a column generation iteration. Moreover, a different approach for find feasible solutions as upper bound can be considered: Solve the integer restricted master problem with CPLEX just in some nodes chosen in a heuristic manner or apply a heuristic method to build a feasible solution. These approaches may be useful to explore the nodes faster, especially in large instances.

We also intend to propose a branch-and-bound algorithm for solving the Model (3.2) and compare its results with the branch-and-price. As techniques based on Lagrangian relaxation are effective for solving the SSCFLP (KLINCEWICZ, 1986; HO, 2015), and as CSP is an extension fo SSCFLP, we intend to implement a solution method based on Lagrangian relaxation for solving CSP too.

REFERENCES

BARNHART, C. et al. Branch-and-price: Column generation for solving huge integer programs. **Operations Research**, INFORMS, v. 46, p. 316–329, 1998.

BEALE, E. M. L.; TOMLIN, J. A. Special facilities in a general mathematical programming system for non-convex problems using ordered sets of variables. In: **International Conference on Operational Research**. London: Tavistock Publications, 1970. p. 447–454.

BUITRAGO, S.; RODRIGUES, E.; ESPIN, D. Global optimization techniques in gas allocation for continuous flow gas lift systems. In: **SPE Gas Technology Symposium**. [S.l.]: Society of Petroleum Engineers, 1996. p. 375–383.

CAMPONOGARA, E. Optimization of lift-gas allocation using dynamic programming. **Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on**, v. 36, n. 2, p. 407–414, Mar 2006.

CAMPONOGARA, E.; CASTRO, M. d.; PLUCENIO, A. The facility location problem: model, algorithm, and application to compressor allocation. In: **IFAC Symposium on Advanced Control of Chemical Processes**. [S.l.: s.n.], 2006. p. 247–252.

CAMPONOGARA, E.; CASTRO, M. P. d.; PLUCENIO, A. Compressor scheduling in oil fields: A piecewise-linear formulation. In: **Automation Science and Engineering**. [S.l.]: IEEE, 2007. p. 436–441.

CAMPONOGARA, E. et al. Compressor scheduling in oil fields: Piecewise-linear formulation, valid inequalities, and computational analysis. **Optimization and Engineering**, v. 12, p. 153–174, 2011.

CAMPONOGARA, E.; NAKASHIMA, P. H. R. Optimizing gas-lift production of oil wells: piecewise linear formulation and computational analysis. **IIE Transactions**, v. 38, n. 2, p. 173–182, 2006.

CAMPONOGARA, E.; NAZARI, L. F.; MENESES, C. N. A revised model for compressor design and scheduling in gas-lifted oil fields. **IIE Transactions**, v. 44, n. 5, p. 342–351, 2012.

CAMPONOGARA, E.; PLUCENIO, A. Column generation for solving a compressor scheduling problem. In: **IEEE Conference on Automation Science and Engineering**. [S.l.]: IEEE, 2008. p. 796–801.

CHEN, C.; TING, C. Combining Lagrangian heuristic and ant colony system to solve the single source capacitated facility location problem. **Transportation Research Part E: Logistics and Transportation Review**, v. 44, p. 1099–1122, nov. 2008.

CONFORTI, M.; CORNUÉJOLS, G.; ZAMBELLI, G. **Integer Programming**. 1. ed. [S.l.]: Springer International Publishing, 2014. 456 p.

CONTRERAS, I.; DÍAZ, J. A.; FERNÁNDEZ, E. Branch and price for large-scale capacitated hub location problems with single assignment. **INFORMS Journal on Computing**, v. 23, p. 41–55, 2010.

DANTZIG, G. B.; WOLFE, P. Decomposition principle for linear programs. **Operations Research**, v. 8, n. 1, p. 101–111, 1960.

DELMAIRE, H.; DÍAZ, J. A.; FENÁNDEZ, E. Reactive grasp and tabu search based heuristics for the single source capacitated plant location problem. **INFOR**, v. 37, p. 194–225, 1999.

DESAULNIERS, G.; DESROSIERS, J.; SOLOMON, M. M. (Ed.). **Column Generation**. Boston, MA: Springer US, 2005. 388 p.

DESROSIERS, J.; SOUMIS, F.; DESROCHERS, M. Routing with time windows by column generation. **Networks**, v. 14, p. 545–565, 1984.

DÍAZ, J. A.; FERNÁNDEZ, E. A branch-and-price algorithm for the single source capacitated plant location problem. **Journal of the Operational Research Society**, v. 53, p. 728–740, jul. 2002.

EISELT, H. A.; MARIANOV, V. (Ed.). **Foundations of Location Analysis**. 1. ed. [S.l.]: Springer US, 2011. 510 p.

ERLENKOTTER, D. A dual-based procedure for uncapacitated facility location. **Operations Research**, INFORMS, v. 26, n. 6, p. 992–1009, 1978.

FRISKE, M. W.; BURIOL, L. S.; CAMPONOGARA, E. A column generation approach for a compressor scheduling problem. In: **Anais do XLVIII SBPO - Simpósio Brasileiro de Pesquisa Operacional**. Porto de Galinas: [s.n.], 2015. p. 2270—-2281.

GILMORE, P. C.; GOMORY, R. E. A linear programming approach to the cutting-stock problem. **Operations Research**, v. 9, p. 849–859, 1961.

GUASTAROBA, G.; SPERANZA, M. A heuristic for BILP problems: The single source capacitated facility location problem. **European Journal of Operational Research**, v. 238, p. 438 – 450, out. 2014.

HAMEDI, H.; KHAMEHCHI, E. A nonlinear approach to gas lift allocation optimization with operational constraints using particle swarm optimization and a penalty function. **Petroleum Science and Technology**, Taylor & Francis, v. 30, p. 775–785, 2012.

HINDI, K. S.; PIEŃKOSZ, K. Efficient solution of large scale, single-source, capacitated plant location problems. **Journal of the Operational Research Society**, v. 50, n. 3, p. 268–274, mar. 1999.

HO, S. C. An iterated tabu search heuristic for the single source capacitated facility location problem. **Applied Soft Computing**, v. 27, p. 169–178, fev. 2015.

HOLMBERG, K.; RÖNNQVIST, M.; YUAN, D. An exact algorithm for the capacitated facility location problems with single sourcing. **European Journal of Operational Research**, Elsevier, v. 113, n. 3, p. 544–559, mar 1999.

KANU, E. P.; MACH, J.; BROWN, K. E. Economic approach to oil production and gas allocation in continuous gas lift. **Journal of Petroleum Technology**, Society of Petroleum Engineers, v. 33, n. 10, p. 1887–1892, out. 1981.

KEHA, A. B.; FARIAS, I. R. de; NEMHAUSER, G. L. Models for representing piecewise linear cost functions. **Operations Research Letters**, v. 32, p. 44–48, jan. 2004.

KHISHVAND, M.; KHAMEHCHI, E. Nonlinear risk optimization approach to gas lift allocation optimization. **Industrial & Engineering Chemistry Research**, American Chemical Society, v. 51, p. 2637–2643, jan. 2012.

KLINCEWICZ, H. L. J. G. A Lagrangian relaxation heuristic for capacitated facility location with single-source constraints. **The Journal of the Operational Research Society**, Palgrave Macmillan Journals, v. 37, n. 5, p. 495–500, maio 1986.

KLOSE, A.; DREXL, A. Facility location models for distribution system design. **European Journal of Operational Research**, v. 162, p. 4–29, abr. 2005.

LASDON, L. S. **Optimization Theory for Large Systems**. [S.l.]: Macmillan, 1970.

LÜBBECKE, M. E.; DESROSIERS, J. Selected topics in column generation. **Operations Research**, v. 53, n. 6, p. 1007–1023, 2005.

MISENER, R.; GOUNARIS, C. E.; FLOUDAS, C. A. Global Optimization of Gas Lifting Operations: A Comparative Study of Piecewise Linear Formulations. **Industrial & Engineering Chemistry Research**, v. 48, p. 6098–6104, 2009.

NAZARI, L. F. **Um Modelo Revisado para Escalonamento de Compressores em Campos de Petróleo: Desiguladades Válidas e Alogirtmo de Planos de Corte**. 103 p. Dissertação (Mestrado) — Universidade Federal de Santa Catarina, 2011.

NISHIKIORI, N. et al. An Improved Method for Gas Lift Allocation Optimization. In: **SPE Annual Technical Conference and Exhibition**. [S.l.]: Society of Petroleum Engineers, 1989.

PADBERG, M. Approximating separable nonlinear functions via mixed zero-one programs. **Operations Research Letters**, v. 27, p. 1–5, ago. 2000.

SAVELSBERGH, M. A branch-and-price algorithm for the generalized assignment problem. **Operations Research**, v. 45, p. 831–841, 1997.

SHERALI, H. D. On mixed-integer zero-one representations for separable lower-semicontinuous piecewise-linear functions. **Operations Research Letters**, Elsevier, v. 28, p. 155–160, 2001.

YANG, Z.; CHU, F.; CHEN, H. A cut-and-solve based algorithm for the single-source capacitated facility location problem. **European Journal of Operational Research**, v. 221, p. 521 – 532, 2012.

APPENDIX : RESUMO EM PORTUGUÊS

No processo de produção de petróleo, ao longo do tempo de extração dos hidrocarbonetos (mistura de petróleo, gás e água), a pressão interna do poço de extração diminui e pode não ser suficiente pra elevar tal mistura até a superfície em quantidade rentável. Para contornar tal situação, são utilizadas técnicas de elevação artificial, tal como o *continuous gas-lift*, ou gás de elevação. Essa técnica consiste em injetar gás sob alta pressão proveniente de compressores até o fundo da tubulação de produção de um poço. O gás entra no tubo de produção através de válvulas e mistura-se aos hidrocarbonetos. Com a gaseificação, ocorre a redução do peso da coluna de fluído e então a mistura consegue ser elevada até a superfície.

Para utilização do gás de elevação em um poço, é necessário um estudo prévio que define o volume e pressão de gás a ser injetado no poço ao longo de sua vida útil. Esses valores são superestimados devido a eventos não-preditivos, com a mudança na natureza física de um poço ou a falha de um compressor. Se um compressor comprimir mais gás que a demanda dos poços que ele atende, existe uma perda de energia, pois o gás excedente deve ser exportado ou queimado (*flaring*), o que acarreta em um aumento nos custos de produção. O problema de escalonamento de compressores (CSP) consiste em alocar um conjunto N de compressores de gás de elevação a um conjunto M de poços de petróleo de forma que os custos de operação sejam minimizados.

Proposto e formulado por Camponogara, Castro e Plucenio (2007), o CSP é uma extensão do problema *Single-Source Capacitated Facility Location Problem*, sendo classificado como um problema misto inteiro não-linear (MINLP). Para a aplicação de técnicas de programação linear inteira ao CSP, é necessário utilizar a linearização por partes, onde é definido um número de pontos sobre a função não-linear, e entre os pontos são traçadas retas. Esse trabalho apresenta 2 tipos de linearizações por partes, conhecida como combinação convexa desagregada (DCC) e combinação convexa através de variáveis do conjunto especial ordenado do tipo II (SOS2). Ambas formulações são denominadas de modelos compactos.

Esse trabalho propõe um algoritmo de *branch-and-price* (B&P) para a resolução do CSP de forma exata. B&P consiste em embutir um algoritmo de geração de colunas (CG) em um *framework* de *branch-and-bound*.

A partir do modelo de linearização por partes, o problema é decomposto em um problema mestre e em subproblemas (um para cada compressor). No problema mestre, as variáveis são associadas a uma atribuição de um compressor a poços, denominada de coluna. O número de colunas para cada compressor é potencialmente exponencial, o que limita muito a resolução de instâncias para o problema. Dessa forma, define-se o problema mestre restrito (RMP), que nada mais é que o problema mestre somente com um sub-conjunto de colunas. A cada iteração o algortimo de geração de colunas resolve a relaxação linear do RMP, e utiliza os valores das variáveis duais para resolver os subproblemas de *pricing*, que utilizam a linearização por partes. O subproblema irá fornecer o custo reduzido de uma nova coluna que não esteja no RMP. Se o custo reduzido for negativo, a coluna pode melhorar a solução do RMP, portanto ela é adicionada ao problema. O algoritmo itera até que nenhuma coluna com custo reduzido negativo seja encontrada, o que significa que a solução ótima relaxada do RMP é ótima também para o problema mestre.

Quando a relaxação do problema mestre restrito é fracionária, é necessário aplicar o B&P, ramificando o problema em dois subproblemas. Para isso, foram definidas duas regras de ramificação que são baseadas nas variáveis da formulação compacta. A primeira regra é denominada *branch (A)*, que consiste em definir um compressor com valor de variável mais próximo da integralidade, e então impor que o compressor seja ativado no ramo à esquerda, e no ramo à direita é imposto que esse compressor seja desativado, *i.e.* ele não atenderá nenhum poço.

Quando a regra *branch (A)* não puder ser aplicada, ou seja, quando nenhum compressor está parcialmente ativado, define-se a segunda regra de ramificação, *branch (B)*. Ela consiste em selecionar uma coluna S_j^b com valor de variável mais próximo da integralidade, e a partir dela definir a regra de ramificação: dado o compressor j^b a qual a coluna S_j^b está associada, e sendo i^b um poço que é atendido nessa coluna, o ramo à esquerda impõem que o compressor j^b atenda ao poço i^b, e o ramo à direita proíbe que o mesmo compressor atenda o poço i^b.

Para a realização dos testes computacionais foram utilizados 3 conjuntos de instâncias, sendo duas delas propostas nesse trabalho, baseadas em instâncias do SSCFLP. Primeiramente foram comparados os valores de limitante inferior (LB) obtidos pela relaxação dos modelos compactos e pelo geração de colunas. Observou-se que o tempo para obtenção do LB através da relaxação do modelo compacto é muito inferior em relação ao necessário para execução do algoritmo de CG. Contudo o LB obtido pelo CG é muito mais apertado, sendo em média inferior a 2%.

Foram realizados testes com a execução dos dois modelos compactos lineares por partes através do *solver* CPLEX. Os resultados mostraram que a utilização da combinação convexa desagregada foi mais eficiente na resolução do CSP, obtendo a solução ótima em menor tempo, bem como qualidade de soluções factíveis sensivelmente melhores quando a solução ótima não foi encontrada.

Em relação ao B&P, foram feitos testes computacionais comparando as duas linearizações por partes utilizadas na resolução do sub-problema de *pricing*. Nesse caso, a utilização de variáveis SOS2 mostrou-se mais eficiente do que a utilização do modelo DCC, obtendo a solução ótima em um tempo de processamento menor.

Por fim, foram comparados os resultados obtidos pelo CPLEX com a linearização por partes DCC e pelo *branch-and-price* com a utilização do sub-problema de *pricing* com variáveis SOS2. Definido um tempo máximo de processamento de 10 horas, CPLEX obteve melhores resultados em relação a tempo para obter a solução ótima do que o B&P no primeiro conjunto de instâncias. Já para o segundo conjunto, o B&P obteve melhores resultados, encontrando a solução ótima em todas as 12 instâncias do conjunto, levando menos tempo que o CPLEX, que por sua vez provou a otimalidade somente em 6 instâncias do conjunto. Em relação ao último conjunto de instâncias, ambas abordagens não puderam encontrar a otimalidade no tempo estipulado. Contudo, a qualidade das soluções inteiras encontradas pelo B&P foi sensivelmente melhor que as soluções encontradas pelo CPLEX. Ainda, para esse último conjunto de instâncias, e para a instância 7 do conjunto 1, o tempo de execução foi estendido para 24h. Com isso, o CPLEX conseguiu provar a otimalidade para a instância 7. Nas demais instâncias não houveram melhoras na solução tanto do CPLEX quanto do B&P, contudo o CPLEX consumiu muita memória, ocasionando a parada abrupta do processo pelo sistema operacional.

A partir dos resultados obtidos, a abordagem de B&P mostrou em média ser mais eficiente que o *solver* CPLEX em relação a tempo para obter a solução ótima, e em relação à qualidade das soluções inteiras encontradas. Como trabalhos futuros pretende-se aperfeiçoar o algoritmo de duas formas: a primeira consiste em utilizar uma abordagem de 2 fases para resolver os sub-problemas de *pricing* (SAVELSBERGH, 1997; BARNHART et al., 1998). A segunda propõe utilizar uma abordagem heurística para obter uma solução factível durante o procedimento de B&P, de forma que os nós sejam explorados mais rapidamente.