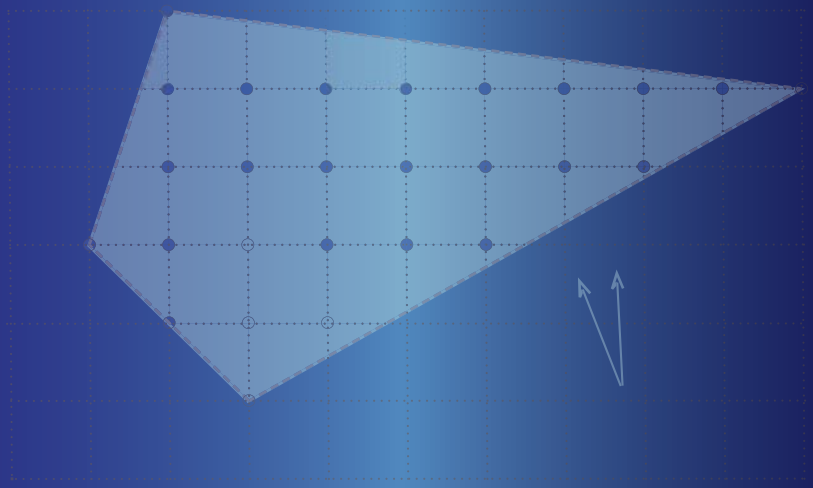
SOLUCIONES ALGEBRAICAS A LA RESOLUCIÓN DE PROBLEMAS MULTIOBJETIVO DISCRETOS

Algebraic Solutions For Solving Discrete Multiobjective Problems

Víctor Blanco Izquierdo



TESIS DOCTORAL

UNIVERSIDAD D SEVILLA

UNIVERSIDAD D SEVILLA

Dpto. de Estadística e Investigación Operativa

Soluciones algebraicas a la resolución de problemas multiobjetivo discretos

Víctor Blanco Izquierdo

Tesis Doctoral

Director: Justo Puerto Albandoz

Sevilla, Marzo 2009

A todos los que han contribuido
en la realización de esta tésis.

UNIVERSIDAD Ð SEVILLA
Departamento de Estadística e Investigación Operativa

Algebraic solutions for solving discrete multiobjective problems

by

Víctor Blanco Izquierdo

Doctoral Dissertation

Advisor: Justo Puerto Albandoz

Sevilla, Marzo 2009

"Ni negro del todo, ni del todo blanco
entre los extremos siempre hay más espacio"
f&f - *Los sueños locos*

Contents

Exordio

La Optimización Multiobjetivo consiste, a grandes rasgos, en calcular los elementos maximales (o minimales) de un conjunto parcialmente ordenado. Este problema ya fue clasicamente tratado por Cantor [25], Cayley [26] y Hausdorff [58] al final del siglo XIX. Sin embargo, el problema de optimización multiobjetivo en sí aparece en un contexto económico en los trabajos de Edgeworth [40] y Pareto [85] tratando de definir un equilibrio económico. Desde entonces, la programación multiobjetivo ha sido fruto de numerosos trabajos de investigación en áreas como la matemática aplicada, la investigación operativa o la economía. Un gran número de libros de texto abordan esta temática, como por ejemplo los libros de Sawaragi, Nakayama y Tanino [92], Chankong y Haimes [28], Yu [113], Miettinen [81], Ehrgott y Figueira [45] o Ehrgott, Figueira y Gandibleux [43; 42].

La importancia de la optimización multiobjetivo no es sólo debida a sus implicaciones teóricas si no también a todas sus aplicaciones prácticas. Muchos problemas reales de toma de decisiones aparecen en la literatura formulados como programas multiobjetivo. Algunos de estos son el diseño de horarios con flujo [62], análisis financieros [45] (Capítulo 20), diseño de redes de transporte [38], problemas de rutas de vehículos [66; 49] u organización de viajes [97].

La mayor parte de los problemas de optimización del mundo real son naturalmente multiobjetivo. Esto es, suelen tener dos o más funciones objetivo que deben satisfacerse simultáneamente y que posiblemente están en conflicto entre sí. Sin embargo, a fin de simplificar su solución, muchos de estos problemas tienden a modelarse como mono-objetivo usando sólo una de las funciones originales y manejando las adicionales como restricciones.

Usualmente, no es posible minimizar simultáneamente todas las funciones objetivo, ya que las funciones objetivo inducen un orden parcial sobre los vectores de la región factible. Así, surge la necesidad de definir un nuevo concepto de solución para este tipo de problemas. Una solución factible será Pareto óptima, no-dominada o de Edgeworth-Pareto si no existe otra solución factible con todos los valores objetivo iguales o menores (con al menos una componente menor estricta).

En esta memoria tratamos problemas multiobjetivo discretos, luego supondremos que las soluciones de estos problemas son vectores con componentes enteras no negativas.

Para resolver problemas multiobjetivo lineales (con todas la funciones involucradas: objetivos y restricciones, lineales) y enteros existen varias metodologías (ver [45]), aunque dos de estás han causado un mayor interés: la enumeración implícita multiobjetivo [115; 116] y la programación dinámica multiobjetivo [110]. Sin embargo, aunque en principio estas pueden ser aplicados a cualquier número de objetivos, mayormente aparecen en la literatura aplicaciones a problemas biobjetivo (dos funciones objetivo). Además de estos métodos generales, existen algunos métodos específicos para resolver problemas biobjetivo que no extienden al caso general. De hecho, en lo que se refiere a algoritmos para resolver problemas multiobjetivo, podemos hablar de dos tipos diferenciados de metodologías atendiendo al número de objetivos: una primera en la que se da el salto de uno a dos objetivos y una segunda, y más profunda, de dos a más de dos objetivos.

Por otra parte, algunos métodos ni siquiera calculan el conjunto completo de soluciones Pareto óptimas, si no sólo las soluciones soportadas (aquellas que pueden obtenerse como soluciones de alguna escalarización lineal del problema multiobjetivo).

En esta memoria se presentan metodologías generales para resolver completamente problemas multiobjetivo lineales y enteros con cualquier número de objetivos.

Aparte de métodos generales, hay algoritmos específicos para problemas combinatorios con dos objetivos: knapsack [111], problema del árbol de caminos mas cortos a mínimo coste [95] o problemas de asignación [87], así como heurísticos y metaheurísticos que disminuyen los tiempos de computación para calcular algunas soluciones del problema.

Sin embargo, la eficiencia computacional no es algo importante a la hora de analizar, con algoritmos exactos, los problemas enteros multiobjetivo pues la mayoría de los problemas multiobjetivo lineales enteros son NP-duros y #P-duros (ver [44] para más detalles). De hecho, existen problemas cuya versión con un sólo objetivo es resoluble en tiempo polinomial mientras que la versión multiobjetivo es NP-dura. Este es el caso de los problemas spanning tree [57] y min-cost flow [41], entre otros. El objetivo de esta memoria es desarrollar herramientas que permitan tratar estos problemas desde su naturaleza intrínseca.

La otra clase de problemas multiobjetivo que abordamos es la de los problemas de optimización polinómica discretos. En estos problemas se supone que

las funciones y las restricciones (de igualdad o desigualdad) son polinómicas y que las variables toman valores enteros no negativos.

La programación polinómica discreta también tiene un gran espectro de aplicaciones. Ejemplos de estas son *gestión de presupuestos* [72], *planificación de la capacidad* [21], *problemas de optimización engrafos* [12], *modelos de selección de carteras con características discretas* [11],
[64] o *ingeniería química* [91], entre muchas otras. Algunas aplicaciones más pueden ser encontradas en [76].

La programación polinómica generaliza la programación lineal y cuadrática y sirve como herramienta para modelar aplicaciones de la ingeniería que se pueden expresar en términos de ecuaciones polinómicas. Incluso problemas con términos trascendentes como senos, logarítmos o radicales pueden reformularse en términos de series de Taylor como problemas polinómicos. En [31] podemos encontrar algunos de los trabajos realizados con respecto a la optimización discreta no lineal.

En esta tesis tratamos desde una perspectiva algebraica los problemas lineales y polinómicos multiobjetivo discretos. En la mayoría de las metodologías que presentamos usamos como herramienta las Bases de Gröbner. Las bases de Gröbner fueron introducidas por Bruno Buchberger en 1965 en su tesis doctoral [23]. Las llamó báses de Gröbner como tributo a su director Wolfgang Gröbner. En principio, estas surgen como una generalización, del caso de una variable al caso multivariado, del máximo común divisor de polinomios.

Una de la aplicaciones posteriores de las bases de Gröbner es en programación lineal y entera (mono-objetivo) a partir del trabajo de Conti y Traverso [30]. A partir de ahí, aparecieron multitud de publicaciones haciendo uso, mejoras o generalizaciones de este trabajo.

Uno de estos trabajos, de Hosten y Sturmfels [60], describe mejoras en el algoritmo de Conti y Traverso que en muchos casos mejora en los procedimientos de ramificación y acotación para resolver problemas enteros de forma exacta. Sin embargo, el lenguaje de estos algoritmos, de ideales tóricos de anillos de polinomios, resultaba difícil de comprender en el área de la optimización, hasta que Thomas presentó en [103] una descripción geométrica e intuitiva de estas metodologías y del procedimiento más conocido para calcular bases de Gröbner: el algoritmo de Buchberger. Otra de las mejoras importantes se da en [105] donde se simplifica el algoritmo de Buchberger para calcular bases de Gröbner cuando se parte de un problema de programación lineal y entero. En [108] se resumen de forma muy clara, para optimizadores, estas metodologías. Algunos de los libros donde también se describen claramente estas metodologías son: Adams y Loustanau [2], Sturmfels [98], Cox et al.

[32] o Bertsimas y Weissmantel [15], y en los artículos de Aardal et al. [1], Sturmfels [99], [100], Sturmfels y Thomas [101] y Thomas [104].

Además, las bases de Gröbner permiten resolver problemas de optimización polinomial basándose en las propiedades de las bases de Gröbner lexicográficas para resolver sistemas de ecuaciones polinómicas (ver [99] para más detalles). Bertsimas et al. [14] presentaron un método para resolver problemas lineales y enteros utilizando esta propiedad. El caso polinómico y continuo se analiza igualmente en el trabajo de Hägglof et al. [56].

Aparte de las bases de Gröbner, en esta tesis usamos también funciones generatrices para resolver problemas multiobjetivo. Barvinok [7] las presenta como una herramienta para contar los puntos enteros dentro de politopos convexos, basandose en los trabajos previos de Brion [22], Khovanskii y Puhlikov [68], y Lawrence [73].

La idea principal de Barvinok es codificar tales puntos enteros en una función racional con tantas variables como la dimensión del espacio. Esto es, si $P \subset \mathbb{R}^d$ es un poliedro convexo, los puntos enteros de P se pueden codificar en la suma formal $f(P,z) = \sum_\alpha z^\alpha$ con $\alpha = (\alpha_1, \ldots, \alpha_d) \in P \cap \mathbb{Z}^d$, y donde $z^\alpha = z_1^{\alpha_1} \cdots z_d^{\alpha_d}$. La aportación de Barvinok consistió en representar tal suma formal de monomios como una suma "corta" de funciones racionales. De hecho, el algoritmo de Barvinok calcula estas funciones en tiempo polinomial para dimensión fija. Un ejemplo claro es el poliedro $P = [0, N] \subset \mathbb{R}$: la expresión larga sería $f(P,z) = \sum_{i=0}^{N} z^i$, mientras que es sobradamente conocido que su representación como suma corta de funciones racionales es $\frac{1-x^{N+1}}{1-x}$.

El uso de funciones generatrices en optimización tampoco es nuevo pues existen algoritmos para resolver, exactamente, problemas de programación lineal y entera, basados en funcione generatrices. En DeLoera y varios [35] se describen cinco métodos distintos para resolver programas lineales y enteros usando las funciones racionales cortas de Barvinok sobre el politopo dado por las restricciones del problema. Además, el grupo de DeLoera ha desarrollado un software, LattE4 [36], dónde se han implementado gran cantidad de algoritmos y aplicaciones de las funciones cortas de Barvinok. En particular, LattE permite calcular las soluciones de un problema de programación entera mono–objetivo usando tales funciones. Recientemente, DeLoera y varios [37] han desarrollado un algoritmo para obtener las soluciones Pareto-óptimas de problemas multiobjetivo lineales y enteros en tiempo polinomial para dimensión y número de objetivos fijos. En esta memoria presentamos una mejora de ese algoritmo en el que la complejidad es polinomial para dimensión fija pero sin ser necesario fijar el número de objetivos.

Esta tesis doctoral estudia algunos de los aspectos algebraicos de la optimización multiobjetivo lineal y polinomial. Primeramente, en el Capítulo 1 se introducen los conceptos básicos necesarios para el desarrollo de los métodos presentados: la presentación del problema multiobjetivo, y el concepto de solución no dominada (o Pareto óptima); algunas definiciones básicas sobre conjuntos parcialmente ordenados (posets); las nociones necesarias sobre la teoría de Bases de Gröbner para ideales polinómicos; y finalmente los resultados más importantes sobre funciones racionales, en especial, para su aplicación a la Programación Lineal y Entera. Los capítulos 2 y 3 están dedicados a la resolución de problemas multiobjetivo lineales y enteros usando bases de Gröbner parciales. En el Capítulo 2 se trata el problema desde una visión totalmente polinómica, presentando los algoritmos sobre anillos de polinomios. Sin embargo, en el Capítulo 3, se presenta una traducción geométrica de los resultados del capítulo anterior. En el Capítulo 4 el mismo problema es abordado usando funciones racionales. En éste se estudia el problema multiobjetivo lineal y entero, y se prueban algunos resultados sobre la complejidad de los métodos que se presentan. Al final del capítulo se describe un método para calcular el número de semigrupos numéricos con genero dado, como aplicación de las funciones generatrices. En el Capítulo 5 se describen distintas metodologías para resolver problemas multiobjetivo polinómicos discretos usando Bases de Gröbner, aprovechando las propiedades de estas para resolver sistemas de ecuaciones polinómicas.

Preface

Multiobjective optimization, in general terms, consists of determining the maximal (minimal) elements of a partially ordered set. This problem was already addressed by Cantor [25], Cayley [26] and Hausdorff [58] at the end of the nineteenth century. However, the multiobjective optimization problem in itself appeared in Economic Theory in the nineteenth century in the seminal papers by Edgeworth [40] and Pareto [85] to define an economic equilibrium. Since then, multiobjective programming (including multicriteria optimization) has been a fruitful research field within the areas of applied mathematics, operations research, and economic theory. Excellent textbooks and surveys papers are available in the literature, the interested reader is referred to the books of Sawaragi, Nakayama and Tanino [92], Chankong and Haimes [28], Yu [113], Miettinen [81] or Ehrgott, Figueira and Gandibleux [42], and to the surveys in [45] and [43].

The importance of multiobjective optimization is not only due to its theoretical implications but also to its many applications. Witnesses of that are the large number of real-world decision problems that appear in the literature formulated as multiobjective programs. Examples of them are *flowshop scheduling* (see [62]), *analysis in Finance* (see [45], Chapter 20), *railway network infrastructure* capacity (see [38]), *vehicle routing problems* (see [66; 49]) or *trip organization* (see [97]) among many others.

Many real world problem are naturally multiobjective, i.e. they use to have two or more objective functions that must be simultaneously optimized, and possibly they are in conflict. However, to simplify the resolution of these problems, many of them are formulated as single-objective problems.

Usually, it is not possible to minimize all the objective functions simultaneously since objective functions induce a partial order over the vectors in the feasible region, so a different notion of solution is needed. A feasible vector is said to be Pareto-optimal, non-dominated or Edgeworth-Pareto solutions if no other feasible vector has componentwise smaller objective values (with at least one component strictly smaller).

In this thesis we study multiobjective discrete problems. First, we treat multiobjective integer linear problems (MOILP). Thus, we assume that both objective functions and constraints that define the feasible region are linear, and that the feasible vectors are non-negative integers.

There are nowadays several exact methods to solve MOILP [45]. Two of them claimed to be of general use and have attracted the attention of researchers over the years: multiobjective implicit enumeration [115; 116] and multiobjective dynamic programming [110]. Neverthless, although in principle they may be applied to any number of objectives, one can only find, in the literature, applications to bicriteria problems. On the other hand, there are several methods that apply to bicriteria problems but that do not extend to the general case. Thus, one can see that there are two thresholds in multiobjective programming, a first step from 1 to 2 objectives and a second, and deeper one, from 2 to more than two objectives. Thus, most of the times, algorithms to solve multiobjective integer problems are designed to compute only the solutions for the bicriteria case.

Moreover, some methods even do not provide the entire set of Pareto-optimal solutions, but the supported ones (those that can be obtained as the solution of a linear scalarization of the multiobjective problem).

Apart from those generic methods, there are specific algorithms for solving some combinatorial biobjective problems: *biobjective knapsacks* [111], *biobjective minimum spanning tree problems* [95] or *biobjective assignment problems* [87], as well as heuristics and metaheuristics algorithms that decrease the CPU time for computing the nondominated solutions for specific biobjective problems.

It is worth noting that most of MOILP problems are NP-hard and intractable [44]. Even in most cases where the single-objective problem is polynomially solvable the multiobjective version becomes NP-hard. This is the case of spanning tree [57] and min-cost flow problems [41], among others.. Therefore, computational efficiency is not an issue when analyzing MOILP. The important point is to develop tools that can handle these problems and that give insights into their intrinsic nature.

The second family of problems that we treat is that of multiobjective polynomial discrete optimization problems. For these problems, we assume that both objective functions and functions that define the constrains if the problem are polynomial, and that the feasible vectors are non-negative integers.

Polynomial programs have a wide spectrum of applications. Examples of them are *capital budgeting* [72], *capacity planning* [21], *optimization problems in graph theory* [12], *portfolio selection models with discrete features* [11; 64]

or *chemical engineering* [91], among many others. The reader is referred to [76] for further applications.

Polynomial programming generalizes linear and quadratic programming and can serves as a tool to model engineering applications that are expressed by polynomial equations. Even those problems with transcendental terms such as sin, log, and radicals can be reformulated by means of Taylor series as a polynomial program. A survey of the publications on general nonlinear integer programming can be found in [31].

In this thesis we treat linear and polynomial multiobjective problems with an algebraic perspective. One of the elements that we use for solving mutiobjective these problems are the Gröbner bases. Gröbner bases were introduced by Bruno Buchberger in 1965 in his PhD Thesis [23]. He named it Gröbner basis paying tribute to his advisor Wolfgang Gröbner. This theory emerged as a generalization, from the one variable case to the multivariate polynomial case, of the greatest common divisor, in an ideal sense. Although, this is only a part of the developed theory from Buchberger's contribution. One of the outcomes of Gröbner Bases Theory was its application to Integer Programming, firstly published by Conti and Traverso [30]. After this paper, a number of publications using Gröbner bases to solve integers programs, appeared in the literature.

In [60], Hosten and Sturmfels gave two ways to implement Conti and Traverso algorithm, that improve in many cases branch-and-bound algorithm to solve, exactly, integer programs. Thomas presented in [103] a geometric point of view of Buchberger algorithm as a method to obtain solutions of an integer program. Later, Thomas and Weissmantel [105] improve Buchberger algorithm in its application to solve integer programs introducing truncated Gröbner basis. At the same time, Urbaniak et al [108], published a clear geometric interpretation of the reduction steps of this kind of algorithms in the original space (decision space). The interested reader can find excellent descriptions of this methodology in the books by Adams and Loustanau [2], Sturmfels [98], Cox et al [32] or Bertsimas and Weissmantel [15], and in the papers by Aardal et al. [1], Sturmfels [99], [100], Sturmfels and Thomas [101] and Thomas [104].

With a different approach of the use of Gröbner bases for solving optimization problems, Bertsimas et al. [14] presented a method for solving linear integer programs, based on the application of Gröbner bases for solving system of polynomial equations. This advantage of Gröbner bases is also used in the paper by Hägglof et al. [56] for solving continuous polynomial optimization problems. Further details about Gröbner bases can be found in [32; 33].

The other tool that we use in this work are the short rational functions. They were initially used by Barvinok [7] as a tool to develop an algorithm for counting the number of integer points inside convex polytopes, based in the previous geometrical papers by Brion [22],Khovanskii and Puhlikov [68], and Lawrence [73]. The main idea is encoding those integral points in a rational function in as many variables as the dimension of the space where the body lives. Let $P \subset \mathbb{R}^d$ be a given convex polyhedron, the integral points may be expressed in a formal sum $f(P,z) = \sum_\alpha z^\alpha$ with $\alpha = (\alpha_1, \ldots, \alpha_d) \in P \cap \mathbb{Z}^d$, where $z^\alpha = z_1^{\alpha_1} \cdots z_d^{\alpha_d}$. Barvinok's aimed objective was representing that formal sum of monomials in the multivariate polynomial ring $\mathbb{Z}[z_1, \ldots, z_n]$, as a "short" sum of rational functions in the same variables. Actually, Barvinok presented a polynomial-time algorithm when the dimension, n, is fixed, to compute those functions. A clear example is the polytope $P = [0, N] \subset \mathbb{R}$: the long expression of the generating function is $f(P,z) = \sum_{i=0}^{N} z^i$, and it is easy to see that its representation as sum of rational functions is the well known formula $\frac{1-z^{N+1}}{1-z}$.

Brion proved in 1988 [22], that for computing the short generating function of the formal sum associated to a polyhedron, it is enough to do it for tangent cones at each vertex of P. Barvinok applied this function to count the number of integral points inside a polyhedron P, that is, $\lim_{z \to (1,\ldots,1)} f(P,z)$, that is not possible to compute using the original expression, but it may be obtained using tools from complex analysis over the rational function f.

The above approach, apart from counting lattice points, has been used to develop some algorithms to solve, exactly, integer programming. Actually, De Loera et al [35] and Woods and Yoshida [112] presented different methods to solve this family of problems using Barvinok's rational function of the polytope defined by the constraints of the given problem.

This doctoral thesis studies some of the algebraic aspects of polynomial and lineal multiobjective discrete optimization. First, in Chapter 1 some background for the methods that have been developed: the multiobjective problem, and the notion of nondominated solution; some basic definitions for partially ordered sets (posets); a brief introduction to Gröbner bases for polynomial ideals; and finally, some results about generating functions, specially, for its application to linear integer programming. In chapters 2 and 3 are dedicated to solve MOILP using partial Gröbner bases. In Chapter 2 the problem is tackled from a polynomial viewpoint, introducing the algorithms for toric ideals. In Chapter 3, the results of the above chapter are translated to a geometrical language. In Chapter 4 the same problem is tackled using generating functions. Here, some complexity results about this approach are proven. At the

end of the chapter we describe a method for counting the number of numerical semigroups with given gender using generating functions. In the last chapter we present some methodologies for solving general polynomil multiobjective integer programs based on the construction of the reduced Gröbner bases of certain ideals related to the problem and on solving triangular systems of polynomial equations given by those bases.

CHAPTER 1

Preliminaries

The objective of this first chapter of preliminaries is to provide the reader with the basic background and the notation used throughout the text. First, the multiobjective programming problem is introduced. The notion of solution considered for this kind of problems is given. Then, a brief introduction to Gröbner bases is presented and finally, the most important results on generating functions are described.

1.1. Multiobjective Optimization

The usual way to present multiobjective optimization problem (MOP) is the following:

$$
\begin{array}{lll}
\min & \{f_1(x), \ldots, f_k(x)\} & \\
s.t. & h_r(x) = b_r & r = 1, \ldots, s \\
& g_i(x) \leq d_i & i = 1, \ldots, m \\
& x \in \mathbb{R}^n &
\end{array} \tag{1}
$$

with $f_1, \ldots, f_k, g_1, \ldots, g_m, h_1, \ldots, h_s$ some functions over $\mathbb{R}^n$ and $b_1, \ldots, b_r$, $d_1, \ldots, d_m$ real numbers.

The functions that want to be minimized, $f_1, \ldots, f_k$, are called the *objective functions* of the problem. The collection of equation, h_r, and inequations, g_i, that have to be satisfied are said the *constraints*, and the space generated by that equations and inequations is called the *feasible region*.

When it is supposed that the variables have to be integer and the objectives functions and restriction are linear expressions, the problem is called *multiobjective linear integer problem* (MOLIP), that is :

$$
\begin{array}{lll}
\min & (c_1\, x, \ldots, c_k\, x) & \\
s.t. & \displaystyle\sum_{j=1}^{n} e_{rj}\, x_j = b_r & i = 1, \ldots, s \\
& \displaystyle\sum_{j=1}^{n} a_{ij}\, x_j \leq d_i & i = 1, \ldots, m \\
& x_j \in \mathbb{Z}_+ & j = 1, \ldots, n
\end{array} \tag{2}
$$

with c_l, a_{ij}, b_r, d_i integers and x_i non negative. From now on, without loss of generality, we will consider the above problem in its standard form, i.e., the coefficient of the objective functions are non-negative and the constraints are in equation form and defining a polytope (bounded).

Indeed, let $A = (a_{ij}) \in \mathbb{Z}^{m\times n}$, $E = (e_{rj}) \in \mathbb{Z}^{s\times n}$, $b = (b_r) \in \mathbb{Z}^s_+$, $d = (d_i) \in \mathbb{Z}^m_+$, and $C = (c_{ij}) \in \mathbb{Z}^{k\times n}$. If C has negative components, set $w_j = \min\{0, \min\{\sum_{i=1}^{n} c_{ij}\, x_i : A\, x \leq d, E\, x = b, x_i \in \mathbb{Z}_+\}\}$ and define new variables $y_j = \sum c_{ij}x_i - w_j$, $j = 1, \ldots, k$. Then, Problem (11) is equivalent to solving:

$$
\begin{array}{lll}
\min & (y_1, \ldots, y_k) & \\
s.t. & & \\
& E\, x & = b \\
& A\, x + Id_m\, x_s & = d \\
& y_j & = \sum_{i=1}^{n} c_{ij}x_i - w_j \quad j = 1, \ldots, k \\
& x_j \in \mathbb{Z}_+ & j = 1, \ldots, n \\
& y_i \in \mathbb{Z}_+ & i = 1, \ldots, k \\
& (x_s)_i \in \mathbb{Z}_+ & i = 1, \ldots, m
\end{array}
$$

where Id_m is the identity matrix of order m. This equivalent formulation has nonnegative coefficients in the objective matrix and constraints in equation form.

Therefore, from now on we deal with $MIP_{A,C}(b)$ in its standard form, which is usually written as:

$$v - \min\{Cx : Ax = b, x \in \mathbb{Z}^n_+\}$$

where $A \in \mathbb{Z}^{m\times n}$, $b \in \mathbb{Z}^m_+$, $C \in \mathbb{Z}^{k\times n}_+$ and $v - \min$ stands for obtaining all the minimal elements in the partially ordered set with the order induced by the matrix C. Then, $MIP_{A,C}$ represents the family of multiobjective problems where the right-hand side varies.

It is clear that the above problem is not a usual optimization problem since the objective function is a vector, thus inducing a partial order among its feasible solutions. Hence, solving the above problem requires an alternative concept of solution, namely the set of non-dominated or Pareto-optimal points (vectors).

A vector $\widehat{x} \in \mathbb{R}^n$ is said to be a *Pareto optimal solution* of $MIP_{A,C}$ if there is no other feasible vector y such that

$$c_j\, y \leq c_j\, \widehat{x} \qquad \forall j = 1, \ldots, k$$

with at least one strict inequality for some j. If x is a Pareto optimal solution, the vector $(c_1\, x, \ldots, c_k\, x)$ is called *efficient*.

We will say that a point, y, is dominated by x if $c_i\, x \lneqq c_i\, y$ for all $i = 1, \ldots, k$, with at least one strict inequality[1]. According to the above concept, solving a multiobjective problem consists of finding its entire set of Pareto optimal solutions.

From the objective function C, we obtain a linear partial order over $\mathbb{Z}^n$ as follows:

$$x \prec_C y :\Longleftrightarrow C\, x \lneqq C\, y.$$

Notice that since $C \in \mathbb{Z}_+^{m \times n}$, the above relation is not complete. Hence, there may exist non-comparable vectors. We will use this partial order, induced by the objective function of Problem $MIP_{A,C}$ as the input for the multiobjective integer programming algorithm developed in this paper.

If the functions involved in a multiobjective program are all polynomials and it is required that the variables are integer numbers, the problem is called *multiobjective polynomial integer program* (MOPIP):

$$(MOPIP_{\mathbf{f},\mathbf{g},\mathbf{h}}) \qquad \begin{array}{lrll} \min & (f_1(x), \ldots, f_k(x)) & & \\ s.t. & g_j(x) & \leq 0 & j = 1, \ldots, m \\ & h_r(x) & = 0 & r = 1, \ldots, s \\ & x & \in \mathbb{Z}_+^n & \end{array}$$

with $f_1, \ldots, f_k, g_1, \ldots, g_m, h_1, \ldots, h_s$ polynomials in $\mathbb{R}[x_1, \ldots, x_n]$ and the constraints defining a bounded feasible region. Therefore, from now on we deal with $MOPIP_{\mathbf{f},\mathbf{g},\mathbf{h}}$ and we denote $\mathbf{f} = (f_1, \ldots, f_k)$, $\mathbf{g} = (g_1, \ldots, g_m)$ and $\mathbf{h} = (h_1, \ldots, h_r)$. If the problem has not equality (resp. inequality) constraints, we denote the problem by $MOPIP_{\mathbf{f},\mathbf{g}}$ (resp. $MOPIP_{\mathbf{f},\mathbf{h}}$), avoiding the nonexistent term.

However, $MOPIP_{\mathbf{f},\mathbf{g},\mathbf{h}}$ can be transformed to an equivalent multiobjective polynomial binary problem as follows: The feasible region $\{x \in \mathbb{R}^n : g_j(x) \leq 0, h_s(x) = 0, j = 1, \ldots, m, r = 1, \ldots, s, x \geq 0\}$, that is assumed to be bounded, can be always embedded in an hypercube $\prod_{i=1}^{n} [0, u_i]^n$, then, every component in x, x_i, has an additional, but redundant, constraint $x_i \leq u_i$. We write x_i in

[1] We are denoting by $\lneqq$ the binary relation "less than or equal to" and where it is assumed that at least one of the inequalities in the list is strict.

binary form, introducing new binary variables z_{ij}:

$$x_i = \sum_{j=0}^{\lfloor log u_i \rfloor} 2^j z_{ij}$$

substituting every x_i in $MOPIP_{\mathbf{f},\mathbf{g},\mathbf{h}}$ we obtain an equivalent $\{0,1\}$-problem.

Alternatively, we can include the polynomials constraints $\prod_{j=0}^{u_i}(x_i - j) = 0$ to restrict x_i to have values in $\{0, 1, \ldots, u_i\}$ to obtain a continuous optimization problem (not binary in this case).

Then, from now on, we will restrict to multiobjective polynomial binary programs ($MOPBP$) in the form:

$$(MOPBP_{\mathbf{f},\mathbf{g},\mathbf{h}}) \qquad \begin{array}{lrll} \min & (f_1(x), \ldots, f_k(x)) & & \\ s.t. & g_j(x) & \leq 0 & j = 1, \ldots, m \\ & h_r(x) & = 0 & r = 1, \ldots, s \\ & x & \in \{0,1\}^n & \end{array}$$

The number of solutions of the above problem is finite, since the decision space is finite. Thus, the number of feasible solutions is, at most $|\{0,1\}^n| = 2^n$.

In this case, a feasible vector $\widehat{x} \in \mathbb{R}^n$ is a nondominated solution of $MOPIP_{\mathbf{f},\mathbf{g},\mathbf{h}}$ if there is no other feasible vector y such that

$$f_j(y) \leq f_j(\widehat{x}) \qquad \forall j = 1, \ldots, k$$

with at least one strict inequality for some j. If x is a nondominated solution, the vector $\mathbf{f}(\mathbf{x}) = (f_1(x), \ldots, f_k(x)) \in \mathbb{R}^k$ is called *efficient*.

Also, a dominated point, y, is dominated by a feasible solution, x, if $f_i(x) \leq f_i(y)$ for all $i = 1, \ldots, k$.

As in the linear case, the objective functions $\mathbf{f} = (f_1, \ldots, f_k)$ induces a partial order on $\mathbb{R}^n$ as follows:

$$x \prec_{\mathbf{f}} y :\Longleftrightarrow \mathbf{f}(x) \lneqq \mathbf{f}(y) \text{ or } x = y.$$

It is clear that this binary relation is reflexive, transitive and antisymmetric. However, notice that since $\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^k$, the above relation is not complete. Hence, there may exist incomparable vectors.

The above order distinguishes solutions with the same objective values and handles them as incomparable. This order can be refined to obtain an antisymmetric partial order over $\mathbb{Z}^n$ as follows:

$$x \preceq_{\mathbf{f}} y :\Longleftrightarrow \begin{cases} \mathbf{f}(x) \lneqq \mathbf{f}(y) & \text{or} \\ \mathbf{f}(x) = \mathbf{f}(y) \text{ and } x \prec_{lex} y & \end{cases}$$

This alternative order allows us rank the solutions those that have the same objective values using the lexicographically order of their components.

Let us consider the following equivalence relation in $\mathbb{Z}^n$:

$$x \sim_C y :\Longleftrightarrow \mathbf{f}(x) = \mathbf{f}(y)$$

The above partial order, $\preceq_C$, allows to solve a simplified version of the multiobjective problem. In this version, we obtain solutions in $\mathbb{Z}^n \setminus \sim_C$, thus having a representative element of each class of nondominated solutions (the lexicographically smallest). With those efficient values, $\{v_1, \ldots, v_t\}$, the remainder solutions can be obtained solving the following polynomial system of equations, in x, for each $i = 1, \ldots, t$:

$$\begin{cases} \mathbf{f}(x) & = v_i \\ \mathbf{g}(x) & = 0 \\ x & \in \{0,1\}^n \end{cases}$$

In special cases, the order $\prec_C$ can be refined to be adapted to specific problems. It is usual to consider slack variables in mathematical programming problems. Two feasible solutions (x, s_1) and (x, s_2), where s_1 and s_2 are the slack component, have the same objective values. Our order, $\prec_{\mathbf{f}}$, considers both solutions as incomparable, but, they are the same, because we are looking just for the x-part of the solution. In these cases, we consider the following refined partial order in $\mathbb{Z}^n \times \mathbb{Z}^r$,

$$(x,s) \prec^s_{\mathbf{f}} (x)(y,s') :\Longleftrightarrow \begin{cases} \mathbf{f}(x) \lneqq \mathbf{f}(y) & \text{or} \\ \mathbf{f}(x) = \mathbf{f}(y) \text{ and } s \prec_{lex} s' \end{cases} \tag{3}$$

where $x, y \in \mathbb{Z}^n_+$ and $s, s' \in \mathbb{Z}^r_+$ represent the slack variables of our problem.

Through this doctoral thesis, we are looking for the entire set of nondominated solutions, and then, unless it is not specifically indicated we use the partial order $\prec_{\mathbf{f}(x)}$.

1.2. Partially ordered sets (posets)

In this section we recall some definitions and constructions related to partially ordered sets as well as extensions of some well-known structures in totally ordered sets to the partial case. They will be useful in the development of the partial theory of Gröbner bases for solving multiobjective problems.

First, we give the basic element to define a partially ordered set: the ordering.

DEFINITION 1.2.1 (Partial ordering). *A partial ordering, $\succ$, in a set S is a binary relation that is reflexive, transitive and antisymmetric, i.e.*

(1) $\forall a \in S$, $a \succ a$. ***Reflexivity***
(2) $\forall a, b, c \in S$, *if* $a \succ b$ *and* $b \succ c$ *then* $a \succ c$. ***Transitivity***
(3) $\forall a, b \in S$, *if* $a \succ b$ *and* $b \succ a$ *then* $a = b$. ***Antisymmetry***

We assume an additional property to be a partial order: $a \succ 0$ for all $a \in S$.

We will use indifferently $\prec$ to denote the opposite binary relation (but also partial ordering) to $\succ$.

A partial order is a *total* order if any two different elements of the set are comparable on the order way, that is, for all $a, b \in S$, either $a \succ b$ or $b \succ a$. An example of partial order in $\mathbb{N}^n$ that is not a total order, is the usual componentwise order on $\mathbb{N}^n$. We will say that a partial order is *strict* if the reflexive condition is not satisfied. A *partial ordered set* (poset) is a pair $(S, \succ)$ where S is a set and $\succ$ is a partial order that orders S.

For x, y in a poset $(S, \succ)$, we write $x \sim y$ if x and y are comparable elements, that is $x \succ y$ or $y \succ x$. If x and y are not comparable, denoted by $x \parallel y$, then they are incomparable. The relation $\sim$ is reflexive and antisymmetric, but not necessarily transitive.

A poset is said locally finite if between each pair of element of it, there is a finite number of elements. $\mathbb{N}$ is locally finite for all orderings over it.

If $(S, \succ)$ is a poset and S a monoid , we say that $\succ$ is *partially admissible* if it is compatible with the sum operation, i.e., for all $a, b, c \in S$, if $a \succ b$ then $a + c \succ b + c$. A total order is a *well order* if any non empty subset of S has a maximal element.

$\mathbb{N}^n$ with the usual vector sum and the scalar product is a ring, and the lexicographic order is an admissible total order over it.

Let $A \subseteq S$, a nonempty set:

- An element $x \in A$ is minimal in A, if for all $y \in A$, $x \prec y$. If A has a unique minimal element then it is called the minimum element of A. If the full poset S has a minimum element, it is called the *bottom element* of S.
- An element $x \in A$ is maximal in A, if for all $y \in A$, $y \prec x$. If A has a unique maximal element then it is called the maximum element of A. If the full poset S has a maximum element, it is called the *top element* of S.

If S is finite, is usual to associate to it a directed graph on this way : for $x, y \in S$, if $x \succ y$, then we draw a line connecting x and y. So, each finite poset can be represented by a $zero - one$ incidence matrix M, where:

$$M_{x,y} = 1 \Leftrightarrow x = y \text{ or } x \succ y, \qquad M_{x,y} = 0 \quad \text{otherwise}$$

The successive powers of M represent the chain connecting each pair of element in S, that is, may be, a pair $x, y \in S$ are not directly comparable, but can exists a way $y \succ y_1 \succ \cdots y_k \succ x$, so, the power $M \overset{k}{\cdots} M$ represents this case.

EXAMPLE 1.2.1 (Total order induced by a matrix). *Let ω in $\mathbb{N}^{n\times m}$. The ω-order, $\prec_\omega$ is defined as:*

$$\alpha \prec_\omega \beta \Leftrightarrow \begin{cases} \omega^1 \alpha \leqslant \omega^1 \beta & or \\ \omega^1 \alpha = \omega^1 \beta, \quad \omega^2 \alpha \leqslant \omega^2 \beta & or \\ \vdots & or \\ \omega^1 \alpha = \omega^1 \beta, \quad \cdots \quad , \omega^{m-1} \alpha = \omega^{m-1} \beta, \quad \omega^m \alpha \leqslant \omega^m \beta \end{cases}$$

Taking $\omega = Id_n(\mathbb{N})$, the induced order is the lexicographical order. In fact, every total and admissible order in M is a ω-order for a specific value of ω [84].
For a fixed vector $c \in \mathbb{N}^n$, we will use the notation c-induced order, $\prec_c$, for the order induced for the matrix:

$$\begin{pmatrix} c \\ Id_n(\mathbb{N}) \end{pmatrix}$$

Therefore, $\prec_c$ is a total order for any $c \in \mathbb{N}^n$.

EXAMPLE 1.2.2 (Partial order induced by a matrix). *Let the monoid $\mathbb{N}^n$, and $\mathbf{\Omega}$ in $\mathbb{N}^{n\times m}$, with rows $\{\Omega_1, \ldots, \Omega_m\}$. The $\mathbf{\Omega}$-order, $\prec_{\mathbf{\Omega}}$ is defined as:*

$$\alpha \prec_{\mathbf{\Omega}} \beta \Leftrightarrow \alpha \prec_{\Omega_i} \beta \text{ for all } i = 1 \ldots, m$$

This order is an admissible partial order on $\mathbb{N}^n$, that generally is not a total order. Taking $\mathbf{\Omega} = Id_n$, is the usually named the componentwise order on M.

1.3. Gröbner Bases

In the following, we give a short description of polynomial algebra giving all necessary definitions and results to present our algorithm.

If $x = (x_1, \ldots, x_n)$ is a list of formal variables, then $x^\alpha = x_1^{\alpha_1} x_2^{\alpha_2} \cdots x_n^{\alpha_n}$, $\alpha_i \in \mathbb{N}$, is called a *monomial* of degree $|\alpha| = \alpha_1 + \alpha_2 + \cdots + \alpha_n$.

Let $\mathbb{R}$ be a field. Finite sums like $\sum_{\alpha \in S \subset \mathbb{N}^n} f_\alpha \, x^\alpha$, where $f_\alpha \in \mathbb{R}$, are called *polynomials* in x over $\mathbb{R}$. The set of polynomials over $\mathbb{R}$ is denoted by $\mathbb{R}[x]$ or $\mathbb{R}[x_1, \ldots, x_n]$. The set of monomials of f is denoted by $mon(f) = \{x^\alpha : \alpha \in S\}$ and the set of monomials in $\mathbb{R}[x_1, \ldots, x_n]$ is denoted by $Mon[x_1, \ldots, x_n]$.

When the number of variables is one, there is an unique way to sort the exponents of monomial, since they are real numbers, and $\mathbb{R}$ is well-ordered. However, if the number of variables is greater than one, n, there is not a unique

way to sort the exponents, since they are in $\mathbb{R}^n$. A fundamental definition for ordering monomials of a polynomial is that of admissible (or monomial) ordering.

A total ordering $\prec$ over $\mathbb{N}^n$ induces an ordering over the monomials in $\mathbb{R}[x_1, \ldots, x_n]$ through the identification $\alpha \mapsto x^\alpha$. From now on, we denote by $\prec$ either the ordering for $\mathbb{N}^n$ or the monomials in $\mathbb{R}[x_1, \ldots, n]$.

If $f = \sum_\alpha f_\alpha \, x^\alpha \in \mathbb{R}[x]$ and $\prec$ is an admissible ordering over $\mathbb{N}^n$, we define the *leading monomial* as $lm(f) = x^\beta$ if $\beta = \max_\prec\{\alpha : x^\alpha \text{monomial in } f\}$, and the *leading coefficient* as $lc(f) = f_\beta$ if $lm(f) = x^\beta$. The *leading term* of f is defined as $lt(f) = lc(f)\, lm(f)$.

An ideal I is a nonempty subset of $\mathbb{R}[x]$ such that

(1) $f, g \in I \Longrightarrow f + g \in I$, and
(2) $f \in \mathbb{R}[x]$ and $g \in I \Longrightarrow f\, g \in I$.

Let $\{f_1, \ldots, f_m\} \subset \mathbb{R}[x]$, the smallest ideal containing $f_1, \ldots, f_m$ is:

$$\langle f_1, \ldots, f_m\rangle = \{\sum_{i=1}^{m} h_i\, f_i : h_i \in \mathbb{R}[x], i = 1, \ldots, m\}$$

The following result states that every ideal can be expressed in the above form:

LEMMA 1.3.1 (Hilbert basis Theorem). *Every ideal I in $\mathbb{R}[x]$ is finitely generated, that is, $I = \langle f_1, \ldots, f_m\rangle$, for some $f_1, \ldots, f_m \in \mathbb{R}[x]$.*

Other fundamental notion in algebraic geometry is the reduction or computation of remainders of the division between polynomials. We describe here how to obtain these remainders.

Given polynomials f and g in $\mathbb{R}[x]$, and $\prec$ an admissible ordering, we write $f \to_g h$ if $h = f - c\, x^\gamma g$ for some $c \in \mathbb{R}$ and some $\gamma \in \mathbb{N}^n$, such that $x^\gamma lm(g) \notin mon(h) = \{x^\alpha : h_\alpha \neq 0\}$.

For a subset $G \in \mathbb{R}[x]$, we write $f \to_G h$ if there exist polynomials $h_1, \ldots, h_{m-1} \in \mathbb{R}[x]$, and $g_1, \ldots, g_m \in G$, such that $f \to_{g_1} h_1 \to_{g_2} \cdots h_{m-1} \to_{g_m} h$.

If moreover h cannot be reduced by any polynomial in G we say that h is the *remainder* or *normal form* of f by G with respect to $\prec$ and is denoted by $nf_G(f)$ or $R(f, G)$.

The following result characterizes the remainder of a polynomial by a set of polynomials with respect to a total admissible ordering. This remainder is computed using the classical extension of the Euclidean division algorithm. A pseudocode for this procedure is shown in Algorithm 1.

Algorithm 1: Usual reduction algorithm

Input: $f \in \mathbb{Z}[x_1, \ldots, x_n]$, $\mathcal{G} = \{g_1, \ldots, g_t\} \subseteq \mathbb{Z}[x_1, \ldots, x_n]$. A total term order, $\prec$, and the polynomials of $\mathcal{G}$ ordered by $\prec$ from the largest to the least.
Output: $R(f, \mathcal{G})$.
Initialization: $R(f, \mathcal{G}_i) = f$
Algorithm: $r = R(f, \mathcal{G})$. While
$\{d \in \{1, \ldots, t\} : lt(g_d)|h \text{ with } g_d \in \mathcal{G}\} \neq \emptyset$:
$r = r - \dfrac{lt(r)}{lt(g_j)}\, g_j$ where
$j = \min\{d \in \{1, \ldots, t\} : lt(g_d)|h \text{ with } g_d \in \mathcal{G}\}$
: $R(f, \mathcal{G}) = r$

PROPOSITION 1.3.1 (Characterization of the Remainder). *Let $f \in \mathbb{R}[x_1, \ldots, x_n]$ and $G = \{g_1, \ldots, g_t\} \subset \mathbb{R}[x_1, \ldots, x_n]$. The above construction of $R(f, \mathcal{G})$ gives a decomposition for f:*

$$f = \sum_{i=1}^{t} f_i\, g_i + r$$

such that, the following properties hold:

(1) *$r = 0$ or no power product that appears in r is divisible by one of the leadings in $\mathcal{G}$.*
(2) *$lt(f) = \max\{r, \max_i\{f_i\, g_i\}\}$.*

The more important definition is the one given by Buchberger in [23]:

DEFINITION 1.3.1 (Gröbner Basis). *G is called a Gröbner basis if every polynomial has an unique normal form with respect to G.*

If F is a finite set of polynomials in $\mathbb{R}[x]$, a Gröbner basis for F is a Gröbner basis, G, such that $\langle F \rangle = \langle G \rangle$.

The most central and original definition towards the algorithm for computing Gröbner bases for any finite set of polynomials, is the concept of S-polynomial.

Let $f, g \in \mathbb{R}[x]$, and choose multiindices α and β such that $x^\alpha\, lm(f) = x^\beta\, lm(g) = LCM(lm(f), lm(g))$ (LMC denotes the least common multiple). Now, define the *S-polynomial* of f and g as:

$$Spol(f, g) = x^\alpha \frac{f}{lc(f)} - x^\beta \frac{g}{lc(g)}$$

THEOREM 1.3.1 (Buchberger). *A finite set G of polynomials is a Gröbner basis if and only if $R(Spol(f, g), G) = 0$, for all $f, g \in G$.*

This result allowed to present an algorithm for computing a Gröbner basis for a given finite set of polynomials in $\mathbb{R}[x]$.

Algorithm 2: Buchberger algorithm

Input : $G := F$
$\prec$ an admissible ordering
Initizalization: $B := \{(f_1, f_2) \in G \times G : f_1 \neq f_2\}$
while $B \neq \emptyset$ **do**
  Choose $(f, g) \in B$;
  $B := B \backslash \{(f, g)\}$;
  if $h := R(Spol(f, g), G) \neq 0$ **then**
    $B := B \cup (G \times \{h\})$;
    $G := G \cup \{h\}$;
  end
end
Output: A Gröbner basis for F: G

If a Gröbner basis for an ideal I in $\mathbb{R}[x_1, x_2, \ldots, x_n]$ is computed relative to the lexicographic ordering with $x_1 \succ x_2 \succ \cdots \succ x_n$, the intersection of I and $\mathbb{R}[x_{l+1}, \ldots, x_n]$, with l from 0 to $n-1$, that we will denote by I_l, is given by the intersection of the Gröbner basis with $\mathbb{R}[x_{l+1}, \ldots, x_n]$. In particular a polynomial f lies in $\mathbb{R}[x_{l+1}, \ldots, x_n]$, if and only if its leading term lies in this subring. This is known as the *elimination property*.

One of the main application of Gröbner bases is the use of these structures for solving systems of polynomial equations with finite number of solutions.

Let $\{f_1, \ldots, f_t\} \subset \mathbb{R}[x_1, \ldots, x_n]$, its *affine variety* is defined as:

$$V(f_1, \ldots, f_t) = \{z \in \mathbb{R} : f_i(z) = 0, \forall i = 1, \ldots, t\}$$

Analogously, let I be an ideal in $\mathbb{R}[x_1, \ldots, x_n]$. We define its affine variety as the set of common roots of all the polynomials in I:

$$V(I) = \{z \in \mathbb{R} : f(z) = 0, \forall f \in I\}$$

By the Hilbert Basis Theorem, every ideal is finitely generated, so it is clear that, if $I = \langle f_1, \ldots, f_t \rangle$, $V(I) = V(f_1, \ldots, f_n)$.

In particular, this gives us a method for solving simultaneous polynomial equations. If there are only finitely many solutions (over an algebraic closure of the field in which the coefficients lie) to the system of equations

$$\begin{cases} f_1(x_1, \ldots, x_n) & = 0 \\ \vdots & \vdots \\ f_m(x_1, \ldots, x_n) & = 0 \end{cases}$$

we should be able to manipulate these equations to get something of the form $g(x_n) = 0$.

The elimination property says that if we compute a Gröbner basis for the ideal generated by $\{f_1(x), \ldots, f_m(x)\}$, relative to the right lexicographic ordering, then we can find the polynomial g as one of the elements of our basis. Furthermore, there will be another polynomial in the basis involving only x_{n-1} and x_n, so we can take our possible solutions for x_n and find corresponding values for x_{n-1}. This lifting continues all the way up until we've found the values of all the variables.

The result to describe this procedure is called *the Extension Theorem*. It can be stated as follows:

THEOREM 1.3.2 (The Extension Theorem). *If $\mathbb{R}$ is algebraically closed, then, a partial solution $(a_{l+1}, \ldots, a_n)$ in $V(I_l)$ extends to $(a_l, a_{l+1}, \ldots, a_n)$ in $V(I_{l-1})$ provided that the leading coefficient polynomials of the elements of a lex Gröbner basis for I_{l-1} do not all vanish at $(a_{l+1}, \ldots, a_n)$.*

These properties allow us to develop an algorithm for solving multiobjective integer programming problems, using Gröbner bases for solving certain system of polynomial equations related to the mathematical program.

1.3.1. Gröbner bases of toric ideals. When the ideals are generated by binomials as follows:

$$I = \langle x^\alpha - x^\beta : A\alpha = A\beta \rangle \subseteq \mathbb{R}[x_1, \ldots, x_n]$$

where $A \in \mathbb{Z}_+^{m \times n}$, such ideal is called *toric*.

There are some methods to apply the toric algebraic geometry to (single-objective) integer programming. All of them have common strategies:

(1) Translate the integer programming problem into a problem about polynomials;
(2) Use the Gröbner bases techniques developed so far to solve the polynomial problem;
(3) Translate the solution of the polynomial problem back into a solution of the integer programming problem.

The first approach to those methods was given by Conti and Traverso [30]. There, from an specific linear integer problem, they build an ideal and the reduced Gröbner basis for it. Then, the optimal solution for the integer problem is the remainder of a certain monomial by the basis. The more important results to go through that method are given here.

First, we consider the following integer programming problem:

$$\begin{array}{lll} \min & c\,x & \\ s.t. & A\,x & = b \\ & x & \in \mathbb{Z}_+^n \end{array} \tag{4}$$

where $c = (c_i) \in \mathbb{Z}_+^n$, $b = (b_j) \in \mathbb{Z}_+^m$ and $A = (a_{ij}) \in \mathbb{Z}_+^{m\times n}$. We denote by $IP_{A,c}(b)$ the above problem and by $IP_{A,c}$ the family of problems where the right-hand side varies.

Introducing the following map $\phi : \mathbb{C}[w_1, \ldots, w_n] \to \mathbb{C}[z_1, \ldots, z_m]$ defined as $\phi(w_j) = \prod_{i=1}^{m} z_i^{a_{ij}}$, and extending to the entire polynomial ring conveniently, the following result is clear.

LEMMA 1.3.2. *A vector $x \in \mathbb{Z}_+^n$ is feasible for Problem (4) if and only if ϕ maps the monomial $w^x = w_1^{x_1} \cdots w_n^{x_n}$ to the monomial $z^b = z_1^{b_1} \cdots z_m^{b_m}$.*

Then, let $f_j = \phi(w_j) = \prod_{i=1}^{m} z_i^{a_{ij}}$ and consider the ideal

$$J_A = \langle f_1 - w_1, \ldots, f_n - w_n \rangle \subset \mathbb{C}[z_1, \ldots, z_m, w_1, \ldots, w_m]$$

With these assumptions, the following result can be stated.

THEOREM 1.3.3. *Algorithm 3 correctly solves Problem (4).*

PROOF. The original proof of this result can be seen in [30]. For an easier understanding, the reader is referred to the explanations given in [2] or [15]. □

Algorithm 3: Conti-Traverso algorithm for single objective IP.

input : $A \in \mathbb{Z}_+^{m\times n}$, $b \in \mathbb{Z}_+^m$, $c \in \mathbb{Z}^{n}{}_+$
output: An optimal solution for Problem (4)
Algorithm:

(1) Compute a Gröbner basis for J_A.
(2) Compute the remainder $g = R(\prod_{i=1}^{m} z_i b_i, G)$.
(3) If $g = w_1^{x_1^*} \cdots w_n^{x_n^*} \in \mathbb{C}[w_1, \ldots, w_n]$, then $(x_1^*, \ldots, x_n^*)$ is an optimal solution to Problem (4). If $g \notin \mathbb{C}[w_1, \ldots, w_n]$, then, Problem (4) is infeasible.

The assumption about the nonnegativity of the objective costs can be done without loos of generality. Furthermore, Algorithm 3 may be modified to allow for A, b having negatives entries. See details of that modification, e.g. [15].

Algorithm 3 raises several computational issues. The Conti and Traverso algorithm has its limitation as the size of A increases since it requires n extra variables over those present in I_A and the Buchberger algorithm for computing Gröbner bases is sensitive to the number of variable involved. A different

algorithm for computing a generating set for I_A without using extra variables can be found in [60].

Once the generating set of I_A has been found, one needs to compute the reduced Gröbner basis of I_A. This can be done by any computer algebra package that does Gröbner basis computation like Maple, Singular, CoCoa, Macaulay2 or Mathematica to name a few. As the size of the problem increases, a straightforward computation of reduced Gröbner bases of I_A can become expensive and even impossible. Several tricks can be applied to help computation, many of which are problem specific.

In the third step of the algorithm, one requires an initial solution to the problem. The original Conti- Traverso algorithm achieves this indirectly during the elimination procedure. Theoretically this task can be as hard as solving the problem, although in practice this depends on the specific problem at hand. The last step, to compute the normal form of a monomial with respect to the current reduced Gröbner basis, is (relatively speaking) a computationally easy task.

In practice, one is, generally, only interested in solving the integer program for a fixed b. In this situation, the Buchberger algorithm can be truncated to produce a sufficient set of binomials that will solve this integer program [105]. This idea was originally introduced in [108] in the context of binary integer programs in which all the data are nonnegative.

A geometric interpretation of Algorithm 3 and more generally of the Buchberger algorithm for toric ideals can be found in [103]. There, a test set for the family $IP_{A,c}$ is a finite subset of vectors in $Ker_{\mathbb{Z}}(A)$ such that for the integer program $IP_{A,c}(b)$ and a non-optimal solution v to this program, there is some u in the test set such that $c\,v > c\,(v-u)$. By interpreting a binomial $x^{\alpha_i} - x^{\beta_i}$ in the Gröbner basis as the vector $\alpha_i - \beta_i \in Ker_{\mathbb{Z}}(A)$, it can be seen that the Gröbner basis is the unique minimal test set for the family $IP_{A,c}$.

Furthermore, in [103] the binomial $\alpha_i - \beta_i$ can also be viewed as the directed line segment $[\alpha_i, \beta_i]$ directed from α_i to β_i. For each $b \in \mathbb{Z}_+^m$ we now construct a directed graph as follows: the vertices of this graph are the solutions to $IP_{A,c}(b)$ and the edges of this graph are all possible directed line segments from the Gröbner basis that connect two vertices of this graph. Then this basis is a necessary and sufficient set of directed line segments such that this graph is a connected graph with a unique sink (at the optimal solution) for each $b \in \mathbb{Z}_+^m$ that makes the problem feasible.

1.4. Short Generating Functions

In this section, we recall some results on generating functions for polytopes, that we use in our development. For details the interested reader is referred to [7; 8; 9].

Let $P = \{x \in \mathbb{R}^n : A\,x \leq b\}$ be a rational polytope in $\mathbb{R}^n$. The main idea of Barvinok's Theory was encoding the integer points inside a rational polytope in a "long" sum of monomials:

$$f(P;z) = \sum_{\alpha \in P \cap \mathbb{Z}^n} z^{\alpha}$$

where $z^{\alpha} = z_1^{\alpha_1} \cdots z_n^{\alpha_n}$.

The following results, due to Barvinok, allow us to re-encode, in polynomial-time for fixed dimension, these integer points in a "short" sum of rational functions.

THEOREM 1.4.1 (Theorem 5.4 in [7]). *Assume n, the dimension, is fixed. Given a rational polyhedron $P \subset \mathbb{R}^n$, the generating function $f(P;z)$ can be computed in polynomial time in the form*

$$f(P;z) = \sum_{i \in I} \varepsilon_i \frac{z^{u_i}}{\prod_{j=1}^{n} (1 - z^{v_{ij}})}$$

where I is a polynomial-size indexing set, and where $\varepsilon \in \{1, -1\}$ and $u_i, v_{ij} \in \mathbb{Z}^n$ for all i and j.

As a corollary of this result, Barvinok gave an algorithm for counting the number of integer points in P. It is clear from the original expression of $f(P;z)$ that this number is $f(P;\mathbf{1})$, but $\mathbf{1} = (1, \ldots, 1)$ is a pole for the generating function, so, the number of integer points in the polyhedron is $\lim_{z \to \mathbf{1}} f(S;z)$. This limit can be computed using residue calculation tools from elementary complex analysis.

Another useful result due to Barvinok and Wood [9], states that computing the short generating function of the intersection of two polytopes, given the respective short generating function for each polytope, is doable in polynomial time.

THEOREM 1.4.2 (Theorem 3.6 in [9]). *Let P_1, P_2 be polytopes in $\mathbb{R}^n$ and $P = P_1 \cap P_2$. Let $f(P_1;z)$ and $f(P_2;z)$ be their short generating functions with at most k binomials in each denominator. Then there exists a polynomial time*

algorithm that computes

$$f(P;z) = \sum_{i\in I} \gamma_i \frac{z^{u_i}}{\prod_{j=1}^{s}(1-z^{v_{ij}})}$$

with $s \leq 2k$, where the γ_i are rational numbers and u_i, v_{ij} are nonzero integral vectors for $i \in I$ and $j = 1, \ldots, s$.

In the proof of the above theorem, the Hadamard product of a pair of power series is used. Given $g_1(z) = \sum_{m\in\mathbb{Z}^d} \beta_m z^m$ and $g_2(z) = \sum_{m\in\mathbb{Z}^d} \gamma_m z^m$, the Hadamard product $g = g_1 * g_2$ is the power series

$$g(z) = \sum_{m\in\mathbb{Z}^n} \eta_m z^m \qquad \text{where } \eta_m = \beta_m\gamma_m.$$

The following Lemma is instrumental to prove Theorem 1.4.2.

LEMMA 1.4.1 (Lemma 3.4 in [9]). *Let us fix k. Then there exists a polynomial time algorithm, which, given functions $g_1(z)$ and $g_2(z)$ such that*

(5)
$$g_1(z) = \frac{z^{p_1}}{(1-z^{a_{11}})\cdots(1-z^{a_{1k}})} \quad \text{and} \quad g_2(z) = \frac{z^{p_2}}{(1-z^{a_{21}})\cdots(1-z^{a_{2k}})}$$

where $p_i, a_{ij} \in \mathbb{Z}^d$ and such that there exists $l \in \mathbb{Z}^l$ with $\langle l, a_{ij}\rangle < 0$ for all i, j, computes a function $h(z)$ in the form

$$h(z) = \sum_{i\in I} \beta_i \frac{z^{q_i}}{(1-z^{b_{i1}})\cdots(1-z^{b_{is}})}$$

with $q_i, b_{ij} \in \mathbb{Z}^d$, $\beta_i \in \mathbb{Q}$ and $s \leq 2k$ such that h possesses the Laurent expansion in a neighborhood U of $z_0 = (e^{l_1}, \ldots, e^{l_n})$ and $h(z) = g_1(z) * g_2(z)$.

For proving Theorem 1.4.2, it is enough to assure that for given polytopes $P_1, P_2 \subseteq Z^n$, their generating functions satisfy conditions (5). It is not difficult to ensure that the conditions are verified after some changes are done in the expressions for the short generating functions (for further details, the interested reader is referred to [9]).

Actually, with this result a general theorem can be proved ensuring that for a pair of polytopes, $P_1, P_2 \subseteq \mathbb{Z}^n$, there exists a polynomial time algorithm to compute, given the generating functions for P_1 and P_2, the short generating function of any boolean combination of P_1 and P_2.

Finally, we recall that one can find, in polynomial time, generating functions for polytopes that are images of polytopes with known generating function.

LEMMA 1.4.2 (Theorem 1.7 in [7]). *Let us fix n. There exists a number $s = s(n)$ and a polynomial time algorithm, which, given a rational polytope $P \subseteq \mathbb{R}^n$ and a linear transformation $T : \mathbb{R}^n \to \mathbb{R}^r$ such that $T(\mathbb{Z}^n) \subseteq \mathbb{Z}^r$, computes the function $f(S; z)$ for $S = T(P \cap \mathbb{Z}^n)$, $S \subseteq \mathbb{Z}^r$ in the form*

$$f(S; z) = \sum_{i \in I} \alpha_i \frac{z^{p_i}}{(1 - z^{a_{i1}}) \cdots (1 - z^{a_i s})}$$

where $\alpha_i \in \mathbb{Q}$, $p_i, a_{ij} \in \mathbb{Z}^r$ and $a_{ij} \neq 0$ for all i, j.

To finish this section, we mention the application of short generating functions to solve single-objective integer programming. The interested reader is referred to [35; 112] for further details.

THEOREM 1.4.3 (Theorem 1 in [35]). *Let $A \in \mathbb{Z}^{m \times n}$, $b \in \mathbb{Z}^m$, $c \in \mathbb{Z}^n$. Assume that m and n are fixed. Rational functions can be used to encode the set of vectors $\{u - v : u$ is an optimal solution, v is a feasible solution, $u, v \in \mathbb{Z}^n\}$, and then solve the MIP problem in time polynomial in the size of the input.*

This result gives an alternative proof of the result by Lenstra [75] where the polynomiallity (for fixed dimension) of integer linear programs is proven.

CHAPTER 2

Partial Gröbner bases: a polynomial approach

The goal of this chapter is to present a new general methodology for solving MOILP using tools borrowed from algebraic geometry. The ideas described here are extensions of some results either for Gröbner bases or its application to solve single objective integer programs.

We present here a methodology to solve exactly multiobjective problems, i.e. providing the whole set of Pareto-optimal solutions (supported and non-supported ones). The approach is done in a polynomial algebra language. An algorithm to compute remainders in the case when the corresponding ordering over the monomials is not total, but partial, is presented. This reduction allows us to extend the concept of Gröbner basis when a partial ordering rather than a total order is considered over $\mathbb{N}^n$. We call these new structures partial Gröbner bases or p-Gröbner bases. We prove that p-Gröbner bases can be generated by a variation of Buchberger Algorithm in a finite number of steps. The main property of p-Gröbner bases of a toric ideal is that, for each element in the ideal, the reduction by maximal chains in the basis is the zero set.

We propose two different approaches to solve multiobjective integer programs. The first method consists of three stages. The first one only uses the constraint matrix of the problem and it produces a system of generators for the toric ideal I_A. In the second step, a p-Gröbner basis is built using the initial basis given by the system of generators computed in the first step. This step requires to fix the objective matrix since it is based on the partial order induced by the objectives. Once the right-hand-side is fixed, in the third step the Pareto-optimal solutions are obtained. This computation uses the new concept of partial reduction of an initial feasible solution by the p-Gröbner basis.

This algorithm extends, to some extent, Hosten-Sturmfels' algorithm [60] for integer programs, in the sense that, if we apply our method to single-objective problems, partial reductions and p-Gröbner bases are the standard notion of reductions and Gröbner bases, respectively.

We also analyze a different methodology based in the original idea by Conti and Traverso [30]. It consists of using the big-M method that results in an increasing number of variables, in order to have an initial system of generators. Moreover, this approach also provides an initial feasible solution. Therefore, the first step in the above algorithm can be ignored and the third step is highly simplified. In any case, our first method (the one extending Hosten-Sturmfels approach) has proved to be more efficient than this second one since computation of p-Gröbner basis is highly sensitive to the number of variables.

Both algorithms have been implemented in MAPLE 10. In this chapter we report on some computational experiments based on two different families of problems with different number of objective functions.

2.1. Elements in the partial theory

Given an admissible total order $\prec$ on $\mathbb{N}^n$, we can order the monomials of any polynomial in the polynomial ring $\mathbb{R}[x_1, \ldots, x_n]$ using the following bijection between monomials and non negative integer vectors:

$$\begin{array}{rccc} \eta: & Mon[x_1, \ldots, x_n] & \longrightarrow & \mathbb{N}^n \\ & x_1^{\alpha_1} \cdots x_n^{\alpha_n} & \mapsto & (\alpha_1, \ldots, \alpha_n) \end{array}$$

Using the same application, we can establish a similar relation, when the order is not total, but partial. The following definitions extend those given for total orderings:

DEFINITION 2.1.1. *Let $\prec$ a fixed partial ordering over $\mathbb{N}^n$, and $f(\mathbf{x}) = \sum_{\alpha \in S \subseteq \mathbb{N}^n} f_\alpha \mathbf{x}^\alpha$ be a polynomial in $\mathbb{R}[x_1, \ldots, x_n]$, we will use the following notation:*

- *$setdeg(f) = \max_\prec\{\alpha : \alpha \in S\}$ (the maximal set of S with respect to $\prec$).*
- *$setlc(f) = \{f_\alpha : \alpha \in setdeg(f)\}$*
- *$setlm(f) = \{\mathbf{x}^\alpha : \alpha \in setdeg(f)\}$*
- *$setlt(f) = \{f_\alpha \mathbf{x}^\alpha : \alpha \in setdeg(f)\}$*

In the case where the order is not total, there is not an unique maximal element of a subset of vectors. Though, it is possible to group any subset of M in comparable subsets, and where its elements can be totally ordered. For each f in $\mathbb{R}[x_1, \ldots, x_n]$:

$$\mathcal{F}(f) := \{(f, h) : h \in setlt(f)\}$$

defining a function $\mathcal{F} : \mathbb{R}[x_1, \ldots, x_n] \to \mathcal{P}(\mathbb{R}[x_1, \ldots, x_n] \times Mon[x_1, \ldots, x_n])$. The key is to consider the same polynomial as a different polynomial of itself, depending of the leading term considered. Denoting by π_1 and π_2 the usual projections of $\mathbb{R}[x_1, \ldots, x_n] \times \mathbb{R}[x_1, \ldots, x_n]$ over $\mathbb{R}[x_1, \ldots, x_n]$, is easy to see that:

$$\pi_1(\mathcal{F}(f)) = \{f\} \text{ and } \pi_2(\mathcal{F}(f)) = setlt(f)$$

This definition can be extended to a set of polynomials as follows: Let $\{f_1, \ldots, f_t\}$ be a set of polynomials in $\mathbb{R}[x_1, \ldots, x_n]$, define:

$$\mathcal{F}(f_1, \ldots, f_t) := \bigcup_{i=1}^{t} \mathcal{F}(f_i) = \{(f_i, h) : h \in setlt(f_i), i = 1, \ldots, t\}$$

If $\prec$ is a partial order over $\mathbb{N}^n$, then we have the relation, $\mathcal{R}$, induced by $\prec$:

$$\alpha \mathcal{R} \beta \Leftrightarrow \alpha \prec \beta \text{ or } \beta \prec \alpha.$$

By definition is reflexive and symmetric. For a fixed partial order and for each $p \in Mon[x_1, \ldots, x_n]$, we denote by C_p the set of comparable monomial with p, i.e.,

$$C_p := \{q \in Mon[x_1, \ldots, x_n] : q \mathcal{R} p\}$$

According to this binary relation, for a set of polynomial $\mathcal{G} = \{g_1, \ldots, g_t\}$, the comparable blocks of $\mathcal{G}$ are going to be the maximal chains of the directed graph whose elements are the leading terms of $\mathcal{G}$ ordered by $\prec$, that is, the ordered paths between maximal and minimal elements in $\{setlt(g_j) : j = 1, \ldots, t\}$. In other words, if we write $\Psi = \{h_j^{k_j}\}_{\substack{j=1\ldots,t \\ k_j=1,\ldots,u_j}}$ where $u_j = \#setlt(g_j)$, $setlt(g_i) = \{h_i^1, \ldots, h_i^{u_i}\}$, and $Max(\Psi) = \{M_1, \ldots, M_Q\}$, $Min(\Psi) = \{m_1, \ldots, m_T\}$, then $\mathcal{G}_i^j$ for $i = 1, \ldots, Q$ and $j = 1, \ldots, T$, is the ordered path (if it exists) that goes from M_i to m_j.

The following basic example shows the easy way to compute the internally comparable blocks of a set of polynomials.

EXAMPLE 2.1.1 (Computing connected blocks of a set of polynomials). *Let* $\mathcal{G} = \{g_1 = x^2y^3 + xy^4, g_2 = y^2 - xy^3 + xy, g_3 = x^3 + x^4y^2, g_4 = x^2y + xy^2, g_5 = xy - x\}$ *and* $\prec$ *the* Ω*-partial order given by the following matrix:*

$$\Omega = \begin{pmatrix} 2 & 1 \\ 3 & 5 \end{pmatrix}$$

$setlt(g_1) = \{x^2y^3, xy^4\}$, $setlt(g_2) = \{xy^3\}$, $setlt(g_3) = \{x^4y^2\}$, $setlt(g_4) = \{x^2y, xy^2\}$ *and* $setlt(g_5) = \{xy\}$. *Then,* $\mathcal{F}(\mathcal{G}) = \{(g_1, x^2y^3), (g_1, xy^4), (g_2, xy^3), (g_3, x^4y^2), (g_4, x^2, y), (g_4, xy^2), (g_5, xy)\}$, *and* $\Psi = \{ x^2y^3, xy^4, xy^3, x^4y^2, x^2y, xy^2, xy\}$ *and* $Max(\Psi) = \{xy^4, x^4y^2\}$, $Min(\Psi) = \{xy\}$.

Figure 2.1 corresponds to the directed graph associated to $\mathcal{G}$, i.e., Hasse Diagram for the leading terms according to the construction of this diagram in $\mathbb{N}^2$, and using η.

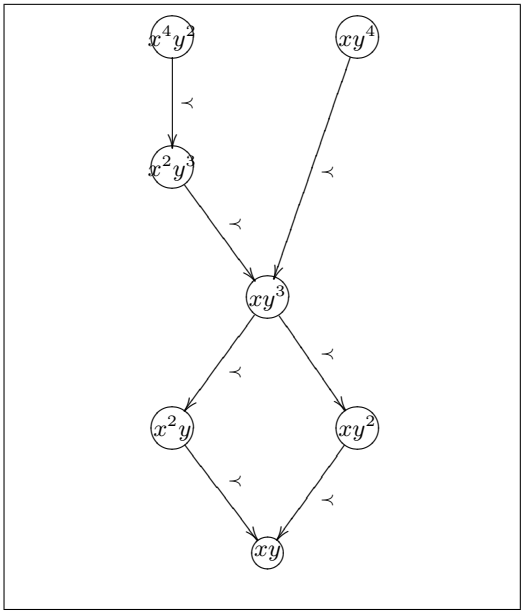


FIGURE 2.1. Hasse diagram of Example 2.1.1

There are four maximal chains:

$\mathcal{G}_1 = \{(g_3, x^4y^2), (g_1, x^2y^3), (g_2, xy^3), (g_4, x^2y), (g_5, xy)\}$
$\mathcal{G}_2 = \{(g_3, x^4y^2), (g_1, x^2y^3), (g_2, xy^3), (g_4, xy^2), (g_5, xy)\}$
$\mathcal{G}_3 = \{(g_1, xy^4), (g_2, xy^3), (g_4, x^2y), (g_5, xy)\}\}$
$\mathcal{G}_4 = \{(g_1, xy^4), (g_2, xy^3), (g_4, xy^2), (g_5, xy)\}\}$

There many different ways to compute the maximal chains of a directed graph, see [5] and [93].

2.2. Gröbner bases for partial orders

In this section an adaptation of the reduction and the Buchberger algorithm when we have an order that is not total, but partial, is presented. This generalization is based on the usual algorithm but it has been thought to be used to get solutions of a multiobjective optimization problem.

2.2.1. Partial remainders. The reduction of the pair (f, h) where $f \in \mathbb{R}[x_1, \ldots, x_n]$ and $h \in setlt(f)$, by an ordered set G with respect to an admissible partial ordering consists of the process described in Algorithm 4. This

Algorithm 4: Partial reduction algorithm by ordered sets.

input : $R = \{(f,h)\}$, $\prec$ an admissible partial order over $\mathbb{N}^n$,
$S = \{(f,h)\}$, $G = \{(g_1,h_1),\ldots,(g_t,h_t)\}$ with $h_{i+1} \prec h_i$.
Set $i := 1$, $S_o = \{\}$.
repeat
  for $(\tilde{f},\tilde{h}) \in S \setminus S_o$ **do**
    while h_i *divides* $\tilde{h}$ **do**
      Set $rm = \tilde{f} - \frac{\tilde{h}}{h_i}\, g_i$ and $R_o = \{rm\} \times setlt(rm)$
      For each $(r,q) \in R_o$ and $(s,p) \in R$:
      if $q \prec p$ **then**
        $R = R \backslash \{(s,p)\}$;
      end
      $S = R_o$.
      $R = R \cup R_o$.
      $S_o = S_o \cup \{(\tilde{f},\tilde{h})\}$.
    end
  end
  $i = i + 1$;
until $i \leq t$;
output: $pR((f,h),G)$, the partial remainder set of (f,h) by G

reduction process extends to the case of a finite collection of ordered sets of pairs by establishing the sequence in which the sets of pairs are considered. Let fix an admissible partial ordering $\prec$. For $f \in \mathbb{R}[x_1,\ldots,x_n]$ and $h \in setlt(f)$ we denote by $pR((f,h),G)_\sigma$ the partial reduction of the pair (f,h) by the family $G = \{G_1,\ldots,G_t\}$ for a fixed sequence of indices $\sigma = (i_1,\ldots,i_t)$.

PROPOSITION 2.2.1. *The above construction of $pR((f,h),G)_\sigma$ for every permutation σ, gives a decomposition for f:*

$$f = \sum_{i=1}^{t} f_i\, g_i + r$$

for all $r \in pR((f,h),G))_\sigma$, $g_i \in G$ and $f_i \in \mathbb{R}[x_1,\ldots,x_n]$ that satisfy yhe following properties:

(1) *$r = 0$ or no power product that appears in r is divisible by one of the leadings monomials in G obtaining a smaller one.*

(2) $h = \max_{\prec}\left\{ r \cap C_h, \max_{\prec}\{p_i\, q_i : p_i \in setlt(f_i) \cap C_h, q_i \in setlt(g_i) \cap C_h\}\right\}$, *where $C_h \subset Mon[x_1,\ldots,x_n]$ is the set of leading monomials comparable to h by $\prec$. (Note that h is uniquely well defined since every element involved to take maximal elements is in the same chain of elements that are comparable with h)*

2.2.2. p-Gröbner bases. Using the above construction of partial reduction, it is possible to give the main definition of this section:

DEFINITION 2.2.1 (p-Gröbner base). *Let $G = \{G_1, \ldots, G_t\}$ be a finite family in $\mathbb{R}[x_1, \ldots, x_n]$ and I the ideal generated by the elements in $\{G_1, \ldots, G_t\}$. G is said to be a partial Gröbner basis (or a p-Gröbner basis for short) of I with respect to an admissible partial order, $\prec$, on $Mon[x_1, \ldots, x_n]$ if and only if, G_i is a maximal chain of G, for each $i = 1, \ldots, t$, and for all $f \in I$ and $h \in setlt(f)$ such that $f \neq 0$ and for all sequences of indices σ, $pR((f, h), G)_\sigma = \{0\}$.*

A p-Gröbner basis is said to be reduced if every element in each maximal chain cannot be obtained by reducing any other element of the same chain.

Given a p-Gröbner basis, computing a reduced p-Gröbner basis is done by deleting the elements that can be reduced by other elements in the basis. After the removing process, the family is a p-Gröbner basis having only non redundant elements. For computing a p-Gröbner basis we describe an algorithm 'a la' Buchberger. Then, we need a definition of Spolynomial for the partial case.

For any two polynomials f_1 and f_2, the Spolynomial with respect to the leading monomials $h_1 = c_1\, x^{\alpha_1} \in setlm(f_1)$, $h_2 = c_2\, x^{\alpha_2} \in setlm(f_2)$ is:

$$\mathbb{S}^k((f_1, h_1), (f_2, h_2)) = \frac{x^\gamma}{h_1}\, f_1 - \frac{x^\gamma}{h_2}\, f_2 \quad k = 1, 2 \tag{6}$$

where $\gamma \in \mathbb{N}^n$ and $\gamma_i = \max\{(\alpha_1)_i, (\alpha_2)_i\}$, $i = 1, \ldots, n$. This definition is exactly the same that in the total (complete order) case, except by the elements where it applies. The main difference will appear when we need to divide these Spolynomial, and then fixing the leading term of them: for $\mathbb{S}^1$ the leading term will be the one with positive coefficient and for $\mathbb{S}^2$ the one with negative coefficient, in the case when both terms are incomparable by $\prec$.

The following lemma is used in the proof of our extended criterion and it is an adaptation of the analogous result for total orders and usual Spolynomials.

LEMMA 2.2.1. *Let $f_1, \ldots, f_s \in \mathbb{R}[x_1, \ldots, x_n]$ be such that there exists $p \in \bigcap_{i=1}^{s} setlm(f_i)$. Let $f = \sum_{i=1}^{s} c_i\, f_i$ with $c_i \in \mathbb{R}$. If there exists $q \in setlm(f)$ such that $q \prec p$, then f is a linear combination with coefficients in $\mathbb{R}$ of Spolynomials of f_i and f_j, $1 \leq i < j \leq s$.*

PROOF. By hypothesis, $f_i = a_i\, p+$ *other smaller or incomparable terms*, with $a_i \in \mathbb{R}$, for all i. Then, f can be rewritten as $f = \sum_{i=1}^{s} c_i\, f_i = \sum_{i=1}^{s} c_i\, a_i\, p+$ *other smaller or incomparable terms.* Since $q \prec p$, then $\sum_{i=1}^{s} c_i\, a_i = 0$.

By definition, for $\mathbb{S}((f_i,p),(f_j,p)) = \frac{1}{a_i}\, f_i - \frac{1}{a_j}\, f_j$, thus,

$$
\begin{aligned}
f &= c_1\, f_1 + \cdots + c_s\, f_s = c_1\, a_1(\frac{1}{a_1}\, f_1) + \cdots + c_s\, a_s(\frac{1}{a_s}\, f_s) \\
&= c_1\, a_1(\frac{1}{a_1}\, f_1 - \frac{1}{a_2}\, f_2) + (c_1\, a_1 + c_2\, a_2)\,(\frac{1}{a_2}\, f_2 - \frac{1}{a_3}\, f_3) + \cdots \\
&+ (c_1\, a_1 + \cdots + c_{s-1}\, a_{s-1})\,(\frac{1}{a_{s-1}}\, f_{s-1} - \frac{1}{a_s}\, f_s) + (c_1\, a_1 + \cdots + c_s\, a_s)\,\frac{1}{a_s}\, f_s \\
&= d_1^k\, \mathbb{S}^k((f_1,p),(f_2,p)) + \cdots + d_{s-1}^k\, \mathbb{S}^k((f_{s-1},p),(f_s,p)) + \left(\frac{1}{a_s}\sum_{i=1}^{s} c_i\, a_i\right) f_s \\
&= \sum_{i=1}^{s-1} d_i^k\, \mathbb{S}^k((f_i,p),(f_{i-1},p)).
\end{aligned}
$$

where $d_i^k = \sum_{j=1}^{i} c_j\, a_j$, for $i = 1,\ldots,s$ and $k = 1,2$. This proves the lemma. □

In the following, we will simplify our notation, whenever it does not cause confusion, and we shall not write the dependence of the leading terms in the $\mathbb{S}$ polynomials. Note that it is possible for maximal chains because their elements only have a leading term.

THEOREM 2.2.1 (Buchberger's Criterion for pGröbner basis). *Let $G = \{G_1,\ldots,G_t\}$ be a finite family of subsets of $\mathbb{R}[x_1,\ldots,x_n]$. The following are equivalent:*

(1) *$G = \{G_1,\ldots,G_t\}$ is a p-Gröbner basis for an ideal I generated by the elements in G.*
(2) *For all $g_i \in G_i, g_j \in G_j$, $h_i \in setlm(g_i)$ and $h_j \in setlm(g_j)$, $pR((\mathbb{S}((g_i,h_i),(g_j,h_j),h),G)_\sigma = \{0\}$ for some σ and for all $h \in setlm(\mathbb{S}((g_i,h_i),(g_j,h_j))$.*

PROOF. If G is a p-Gröbner basis for I, then as $\mathbb{S}((g_i,h_i),(g_j,h_j))$ is in I, and by definition of pGröbner basis, $pR((\mathbb{S}((g_i,h_i),(g_j,h_j),h),G)_\sigma = \{0\}$ for any σ.

Then, by hypothesis f can be written as a linear combinations of the elements in $\mathcal{G}^*$ (this representation is not unique):

$$f = \sum_{i=1}^{d} p_i \, g_i^*$$

for some $p_i \in \mathbb{R}[x_1, \ldots, x_n]$ for $i = 1, \ldots, d$.

Let $X = \{X_1, \ldots, X_N\}$ be the set of maximal elements of the set $\{P_i \, R_i :$
$P_i \in setlm(p_i)$,
$R_i \in setlm(g_i^*)\}$ with respect to $\prec$.

If $X \supseteq setlm(f)$, the polynomial f can be partially reduced by the elements in $\mathcal{G}^*$. This proves the result.

Otherwise, there must exist $l \in setlm(f) \backslash X$. We will prove by contradiction that this case is not possible. Indeed, if $l \in setlm(f)$, it must come from some simplification (reduction) of the linear combination defining f. Then, the construction ensures that it must exist at least one element, $X_i \in X$, such that $l \prec X_i$.

Set $J(X_i) = \{j : P_j \, R_j = X_i \text{ with } P_j \in setlm(p_j), R_j \in setlm(g_j^*)\}$. For any $j \in J(X_i)$, we can write $p_j = P_j + \textit{ other terms}$ and define $q = \sum_{j \in J(X_i)} P_j \, g_j^*$. Then, $X_i \in setlm(P_j \, g_j^*)$, for all $j \in J(X_i)$. However, by hypothesis there exists $Q \in setlm(q)$, with $Q \prec X_i$.

Hence, by Lemma 2.2.1, there exist $d_{s,r}^k \in \mathbb{R}$, $k = 1, 2$, such that:

$$q = \sum_{r,s \in J(X_i), r \neq s, g_s^*, g_r^* \in \mathcal{G}^*} d_{s,r}^k \, \mathbb{S}^k((P_s \, g_s^*, L_s), (P_r \, g_r^*, L_s)) \quad k = 1, 2.$$

for some $L_j \in setlm(P_j \, g_j^*)$ for all $g_j^* \in \mathcal{G}^*$.

Now, for any r, $s \in J(X_i)$, we have that $X_i = lcm(L_r, L_s)$ for some $L_r \in setlm(P_r \, g_r^*)$ and $L_s \in setlm(P_s \, g_s^*)$ and therefore we can write:

$$\begin{aligned} \mathbb{S}^k((P_r \, g_r^*, L_r), (P_s \, g_s^*, L_s)) &= \frac{X_i}{L_r} \, P_r \, g_r^* - \frac{X_i}{L_s} \, P_s \, g_s^* \\ &= \frac{X_i}{l_r} \, g_r^* - \frac{X_i}{l_s} \, g_s^* = \frac{X_i}{P_{r,s}} \, \mathbb{S}^k((g_r^*, l_r), (g_s^*, l_s)) \end{aligned}$$

where $l_r = \frac{L_r}{P_r}$, $l_s = \frac{L_r}{P_s}$, $P_{r,s} = lcm(l_r, l_s)$ and $k = 1, 2$.

By hypothesis, $pR\big(\mathbb{S}^k((g_r^*, l_r), (g_s^*, l_s)), \mathcal{G}^*\big) = \{0\}$. Thus, from the last equation we deduce that:

$$pR(\mathbb{S}^k((P_r \, g_r^*, L_r), (P_s \, g_s^*, L_s)), \mathcal{G}) = \{0\}$$

this gives a representation:

$$\mathbb{S}^k((P_r \, g_r^*, L_r), (P_s \, g_s^*, L_s)) = \sum_{g_\nu^* \in \mathcal{G}^*} p_{r,s}^{k,\nu} \, g_\nu^*$$

with $p_{r,s}^{k,\nu} \in \mathbb{R}[x_1, \ldots, x_n]$ and $k = 1, 2$.

Then, $\{P_{r,s}^{k,\nu}\, R^{\nu}\colon\ g_{\nu}^{*} \in \mathcal{G}^{*}, P_{r,s}^{k,\nu} \in setlm(p_{r,s}^{k,\nu}), R^{\nu} \in setlm(g_{\nu}^{*})$ and do not exist $P_{r,s}^{k,\tilde{\nu}}$ and $R^{\tilde{\nu}}$ satisfying $P_{r,s}^{k,\tilde{\nu}} \in setlm(p_{r,s}^{k,\tilde{\nu}})$, $R^{\tilde{\nu}} \in setlm(g_{\tilde{\nu}}^{*})$ such that $P_{r,s}^{k,\nu}\, R^{\nu} \prec P_{r,s}^{k,\tilde{\nu}}\, R^{\tilde{\nu}}\} = setlm(\mathbb{S}^{k}(P_r\, g_r^{*}, P_s\, g_s^{*}))$.

To simplify the notation, denote $S_{r,s}^{k} = setlm(\mathbb{S}^{k}(P_r\, g_r^{*}, P_s\, g_s^{*}))$.

By construction of Spolynomial, we have that there exists $p \in S_{r,s}^{k}$ such that $p \prec X_i$, so, substituting these expressions into q above, we have

$$\begin{aligned} f &= \sum_{j \notin J(X_i)} p_j\, g_j^{*} + \sum_{j \in J(X_i)} p_j\, g_j^{*} = \sum_{j \notin J(X_i)} p_j\, g_j^{*} + q \\ &= \sum_{j \notin J(X_i)} p_j\, g_j^{*} + \sum_{r,s} d_{r,s}^{k}\, \mathbb{S}^{k}((P_s\, g_s^{*}, L_s), (P_r\, g_r^{*}, L_r)) \\ &= \sum_{j \notin J(X_i)} p_j\, g_j^{*} + \sum_{r,s} \sum_{\nu} p_{r,s}^{k,\nu}\, g_{\nu}^{*}. \end{aligned}$$

Thus, we have expressed f as:

$$f = \sum_{i=1}^{d} p_i'\, g_i^{*}$$

with one leading term, p, smaller than X_i. However, this is a contradiction and the theorem is proved. □

This criterion allows us to describe an algorithm to compute p-Gröbner bases. The pseudocode of that procedure is described in Algorithm 5.

Algorithm 5: Buchberger's algorithm for pGröbner bases.

Input : $F = \{g_1, \ldots, g_k\} \subseteq \mathbb{R}[x_1, \ldots, x_m]$, $I = \langle F \rangle$ and $\prec$ an admissible partial order. $\{F_k\}_{k=1}^{N}$ the partition of $\mathcal{F}(F)$ in blocks of comparable elements.

Initialization: $G_k = F_k$ for all k.

for $p \in G_k$, $q \in G_l$ **do**

  repeat

    if $pR((\mathbb{S}^{k}((p, h_1), (q, h_2)), h), G) = \{0\}$ *for some* $h \in setlm(\mathbb{S}^{k}(p, q))$ **then**

      $G := G \cup \{r\}$ for all $r \in pR((\mathbb{S}^{k}((p, h_1), (q, h_2)), h), G) = \{0\}$ that is not in G

    end

    Part G in maximal chains.

  until $pR((\mathbb{S}^{k}((p, h_1), (q, h_2)), h), G) = \{0\}$ *for all* $p \in G_k, q \in G_l$. ;

endfor

Output: $G = \{G_1, \ldots, G_t\}$ pGröbner basis for I.

By construction, as for the standard Gröbner bases, we can ensure that Algorithm 5 ends. For standard Gröbner bases, it follows from the fact that polynomial rings over a field are noetherian. In the partial case, the same result can be applied taking into account that at each step, instead of one polynomial, several polynomials may be added. The other fact in this proof is that of the notion of *initial ideal* of a given polynomial ideal (further details can be seen in [2]). The extension of this monomial ideal can be defined as the *partial initial ideal* of an ideal $I \subseteq \mathbb{R}[x_1, \ldots, x_n]$ with respect to a partial order, $\prec$, as $p - in_{\prec}(I) = \{h : \exists f \in I \text{ with } h \in setlt(f)\}$.

Furthermore, using the notion of Universal Gröbner basis (see [100] for further details), the p-Gröbner basis is (without taking into account the partition in maximal chains but only its elements) at most as big as the Universal Gröbner basis .

PROPOSITION 2.2.2. *After a finite number of iterations, Algorithm 5, computes a p-Gröbner basis for* $I = \langle F \rangle$.

COROLLARY 2.2.1. *Let* $G = \{G_k\}_{k=1}^{t}$ *be a pGröbner basis for an ideal* $I \subseteq \mathbb{R}[x_1, \ldots, x_n]$. *A reduced pGröbner basis can be computed from* G *as follows:*

(1) *For each* $k = 1, \ldots, t$, *remove from* G_k *all* g_i^k *for which there exists* $j \neq i$ *such that the unique element in* $\pi_2(\mathcal{F}(g_j^k) \cap G_k)$ *divides the unique element in* $\pi_2(\mathcal{F}(g_i^k) \cap G_k)$, *and divide each remaining* g_i^k *by its leading coefficient.*

(2) *Compute the following reduction process for each* $G_k = \{g_1^k, \ldots, g_{i_k}^k\}$*:*
For $j = 1, \ldots, i_k$*:*
Compute $P_j := pR(g_j^k, G_k \backslash \{g_j^k\})$. *If there exists an element in* P_j, h_j, *such that it is comparable with every element in* G_k*:* $G_k := G_k \backslash \{g_j^k\} \cup \{h_j\}$

2.3. Application to integer multiobjective programming

In the following, we present algorithms to solve multiobjective problems analogous to the methods that solve the single objective case, using standard Gröbner basis.

The following lemma states the shape of a Gröbner basis for toric ideals. The partial Gröbner bases Theory developed above will be applied to toric ideals for solving multiobjective linear integer programs.

In the following, we will simplify our notation, whenever it does not cause confusion, and we shall not write the leading terms of monomials since each the leading term of a monomial coincides with itself.

LEMMA 2.3.1. *Let $G = \{G_1, \dots, G_t\}$ be a p-Gröbner basis for $I_A = \langle x^u - x^v : Au = Av\rangle$ with respect to the partially admissible ordering $\prec$. Then, every element in G_i, $i = 1, \dots, t$, is a binomial.*

PROOF. It is clear taking into account that the Spolynomials of binomials are binomials by construction, and that the remainders of binomials by binomials are also binomial. Then, every element that is added to the basis with the Buchberger algorithm is a binomial. □

Methods to solve multiobjective problems using p-Gröbner basis are based on computing the reduction of a feasible solution by the partial basis of a toric ideal. The key for that result is the fact that the partial remainder of any pair of feasible solutions is the same, therefore the algorithm is valid for any initial feasible solution. After the following theorem, Lemma 2.3.1 ensures the same statement for the multiobjective case and p-Gröbner bases.

THEOREM 2.3.1. *Let G be the reduced p-Gröbner basis for $I_A = \langle x^u - x^v : Au = Av\rangle$ and $\alpha \in \mathbb{Z}_+^n$. Then, $pR(x^\alpha, G)_\sigma = pR(x^\alpha, G)_{\sigma'}$, for any sequences σ and σ'.*

PROOF. We first observe that the elements in $pR(x^\alpha, G)_\sigma$ are monomials. Indeed, since the first step of Algorithm 4 reduces the element x^α by a binomial in G, $(x^{\alpha_1} - x^{\beta_1}, x^{\alpha_1})$, then $r = x^\alpha - \frac{x^\alpha}{x^{\alpha_1}}(x^{\alpha_1} - x^{\beta_1}) = x^{\alpha - \alpha_1 + \beta_1}$. Then, the remainders are all monomials.

On other hand, let x^β be an element in $pR(x^\alpha, G)_\sigma$, then $x^\alpha - x^\beta \in I_A$, so $\alpha - \beta \in Ker(A)$ and by Definition 2.2.1, $pR((x^\alpha - x^\beta, x^\alpha), G)_{\sigma'} = pR((x^\alpha - x^\beta, x^\beta), \mathcal{G})_{\sigma'} = \{0\}$ for any σ'. So, $x^\beta \in pR(x^\alpha, G)_{\sigma'}$. □

The above result ensures that without loss of generality remainders of monomials by p-Gröbner bases are independent of the permutation of indices used. Therefore, we do not make reference to σ in the notation, referring always to the natural sequence $\sigma = (1, \dots, t)$.

LEMMA 2.3.2. *Let G be the reduced p-Gröbner basis for I_A and $\alpha_1, \alpha_2 \in \mathbb{Z}_+^n$ such that $A\,\alpha_1 = A\,\alpha_2$. Then, $pR(x_1^\alpha, G) = pR(x_2^\alpha, G)$.*

PROOF. Let $x^\beta \in pR(x_1^\alpha, G)$, then since $A\alpha_1 = A\alpha_2$, $A\,\beta = A\,\alpha_2$. Next, since x^β cannot be reduced because it is in the remainder set, then $x^\beta \in pR(x_2^\alpha, G)$. □

The following theorem states how to solve MOILP using partial Gröbner bases of toric ideals. First, we need to fix the partially admissible ordering

over the monomials in $\mathbb{R}[x_1, \ldots, x_n]$ to compute the basis. We use the partially admissible ordering induced by the objective function as follows:

$$x^u \prec_C x^v :\Longleftrightarrow C\,u \lneqq C\,v.$$

THEOREM 2.3.2. *Let $A \in Z_+^{m\times n}$, $C \in \mathbb{Z}_+^{k\times n}$ and $b \in \mathbb{Z}_+^m$. If $G = \{G_1, \ldots, G_t\}$ is the reduced p-Gröbner basis for the toric ideal: $I_A := \langle x^u - x^v : A\cdot u = A\cdot v\rangle$ with the partial order defined by the cost matrix C, then, the set of exponents of the monomials in $pR(x^\alpha, G)$ coincides with the set of Pareto-optimal solutions for $MIP_{A,C}(b)$.*

PROOF. Let $pR(x^\alpha, G) = \{x^{\alpha_1}, \ldots, x^{\alpha_r}\}$ the set of remainders obtained reducing the *feasible* monomial by a p-Gröbner basis. If β is a feasible solution that dominated α_i, for some $i = 1, \ldots r$, then, $x^{\alpha_i} - x^\beta$ is clearly in I_A since $A\alpha_i = A\beta = b$. By Definition 2.2.1, there is some G_j in G and an element $g_j \in G_j$ such that the leading term of $x^{\alpha_i} - x^\beta$, namely x^{α_i}, can be reduced by g_j, but x^{α_i} has been already reduced because it is in the partial remainder set of x^α. It is a contradiction.

Suppose now that there exists a Pareto optimal solution, α^* that is different of all α_i. Then, $x^\alpha - x^{\alpha^*}$ is in I_A, and by definition of pGröbner basis and because α^* is a Pareto optimal solution, it cannot exists any feasible solution smaller than α^*. Then, since x^α is reduced to the elements in $\{x_1^\alpha, \ldots, x_r^\alpha\}$, there exists j such that $x^{\alpha_j} - x^{\alpha^*}$ is in $pR(x^\alpha - x^{\alpha^*}, G)$, since, x^{α^*} must be contained in $pR(x^\alpha, G)$ by construction. □

The above results states that once we have built a p-Gröbner basis for the toric ideal I_A and an initial feasible solution is given, reducing that solution by that basis it is possible to obtain any Pareto-optimal solution of the problem. However, in general, computing feasible solutions and Gröbner bases is not easy. One way to compute a partial Gröbner basis is to use a system of generators for the ideal. Even in the toric case, in general, it is not true that given a basis, $\mathcal{B}_A$, for $Ker_\mathbb{Z}(A)$, $\mathcal{B}_A$, the toric ideal is generated by $\{x^{u^+} - x^{u^-} : u \in \mathcal{B}_A\}$. Moreover, obtaining a feasible solution, in general, may be as difficult as solving the problem since it consists of solving a system of diophantine equations.

Two methods are described below for solving MOILP. A first one extending the ideas of Conti and Traverso for the single-objective case and another one based on the extension of the improvement given by Hosten and Sturmfels.

The first approach to compute a p-Gröbner basis for a family of multiobjective programs is based on Conti and Traverso method for the single

objective case [30]. For this algorithm, the key is transforming the given multiobjective program into another one where computations are easier and so that an initial set of generators for I_A is known.

Notice that finding an initial set of generators for I_A can be done by a straightforward modification of the Big-M method [10].

Given the program $MIP_{A,C}(b)$, we consider the associated extended multiobjective program, $EMIP_{A,C}(b)$ as the problem $MIP_{\widetilde{A},\widetilde{C}}(b)$ where

$$\widetilde{A} = \left(\begin{array}{c|c|c} & -1 & \\ Id_m & \vdots & A \\ & -1 & \end{array}\right) \in \mathbb{Z}^{m\times(m+1+n)},$$

$\widetilde{C} = (M\cdot\mathbf{1}|C) \in \mathbb{Z}^{(m+1+n)\times k}$, Id_m stands for the $m\times m$ identity matrix, M is a large constant and $\mathbf{1}$ is the $(m+1)\times k$ matrix whose components are all 1. This problem adds $m+1$ new variables, whose weights in the multiobjective function are big, and so, solving this extended minimization program allows us to solve directly the initial program $MIP_{A,C}$. Indeed, any feasible solution to the original problem is a feasible solution to the extended problem with the first m components equal to zero, so any feasible solution of the form $(0,\overset{m+1}{\ldots},0,\alpha_1,\ldots,\alpha_n)$ is non-dominated, upon the order $\prec_{\widetilde{C}}$, by any solution without zeros in the first m components. Then, computing a p-Gröbner basis for the extended program using the partial Buchberger Algorithm, allows us detecting infeasibility of the original problem. Furthermore, a trivial feasible solution, $\widetilde{\mathbf{x}}_0 = (b_1,\ldots,b_m,0,\overset{n+1}{\ldots},0)$, is known and the initial set of generators for $I_A = \langle z^{\alpha_1}w^{\beta_1} - z^{\alpha_2}w^{\beta_2} : \widetilde{A}(\alpha_1,\beta_1) = \widetilde{A}(\alpha_2,\beta_2)$ is given by $\{\prod_{i=1}^{m} z_i^{ai1} - w_1,\ldots,\prod_{i=1}^{m} z_i^{ain} - w_n\}$ (see [2] for further details). Here the z-variables are used for the new slack variables of the extended problem and w for the original variable of the problem.

With these considerations, let $G = \{G_1,\ldots,G_t\}$ be a p-Gröbner basis for the toric ideal $I_{\tilde{A}} = \langle\prod_{i=1}^{m} z_i^{a_{i1}} - w_1,\ldots,\prod_{i=1}^{m} z_i^{a_{in}} - w_n\rangle$ and the admissible partial order induced by C. Then, Algorithm 6 solves correctly $MIP_{A,C}(b)$.

Hoşten and Sturmfels [60] improved the method by Conti and Traverso to solve single-objective programs using standard Gröbner bases. Their improvement is due to the fact that it is not necessary to increase the number of variables in the problem, as Conti and Traverso's algorithm does. Hoşten and Sturmfels's algorithm allows decreasing the number of steps in the computation of the Gröbner basis, but on the other hand, it needs an algorithm to

Algorithm 6: Conti-Traverso algorithm for solving MOILP.

input : $A \in Z_+^{m\times n}$, $C \in \mathbb{Z}_+^{k\times n}$ and $b \in \mathbb{Z}_+^m$.

Set $I_{\tilde{A}} = \langle \prod_{i=1}^{m} z_i^{a_{i1}} - w_1, \ldots, \prod_{i=1}^{m} z_i^{a_{in}} - w_n \rangle$.

Step 1.: Compute a p-Gröbner basis, G, for $I_{\tilde{A}}$, with the ordering induced by C.

Step 2.: Compute the set of partial remainders, $pR(\prod_{i=1}^{m} z_i^{b_i}, \mathcal{G})$.

Step 3.: For each $g^* \in PR(\prod_{i=1}^{m} z_i^{b_i}, G) \cap \mathbb{C}[w_1, \ldots, w_n]$, with $g = w_1^{x_1^*} \cdots w_n^{x_n^*}$, x^* is a Pareto optimal solution for the program.

Step 4.: If $g^* \notin \mathbb{R}[w_1, \ldots, w_n]$ for all g^* in $pR(\prod_{i=1}^{m} z_i^{b_i}, G)$, the problem is infeasible.

compute an initial feasible solution, which was trivial in the Conti and Traverso algorithm. We have modified this alternative algorithm to compute the entire set of Pareto-optimal solutions. The first step in the algorithm is computing an initial basis for the polynomial toric ideal $I_A = \langle x^u - x^v : Au = Av \rangle$. This step does not depend on the order induced by the objective function, so it can be used to solve multiobjective problems. Details can be seen in [60]. Algorithm 7 implements the computation of the set of generators of I_A. This procedure uses the notion of LLL-reduced basis (see [74] for further details). In addition, we use a ω-graded reverse lexicographic term order, $\prec_\omega^{gr_i}$, induced by $x_{i+1} > \cdots > x_{i-1} > x_i$ (with $x_{n+1} := x_1$), that is defined as follows:

$$\alpha \prec_\omega^{gr_i} \beta :\Longleftrightarrow \left\{ \sum_{j=1}^{n} \omega_j \alpha_j < \sum_{j=1}^{n} \omega_j \beta_j \text{ or } \sum_{j=1}^{n} \omega_j \alpha_j = \sum_{j=1}^{n} \omega_j \beta_j \text{ and } \alpha \prec_{lex} \beta \right.$$

where $\omega \in \mathbb{R}_+^n$ is chosen such that $x_{i+1} > \cdots > x_{i-1} > x_i$. Finally, for any $a \in \mathbb{R}$ we denote by $a_+ = \max\{a, 0\}$ and $a_- = -\min\{a, 0\}$. I_A consists of binomials $x^{u_i} - x^{v_i}$ with $u_i - v_i \in Ker(A)$, for $i = 1, \ldots, s$. We compute in the next step a partial Gröbner basis from the initial set $\{x^{u_1} - x^{v_1}, \ldots, x^{u_s} - x^{v_s}\}$ using our extended Buchberger algorithm.

Once we have obtained the partial Gröbner basis, we can compute the entire set of Pareto-optimal solutions for $MIP_{A,C}(b)$ by Algorithm 8.

To illustrate the above approach, we solve an example of MOILP with two objectives where all the computations are done in full detail.

Algorithm 7: `setofgenerators(`A`)`

input : $A \in \mathbb{Z}^{m\times n}$

(1) Find a lattice basis $\mathcal{B}$ for $Ker(A)$ (using the Hermite Normal Form).

(2) Replace $\mathcal{B}$ by the LLL-reduced lattice basis $\mathcal{B}_{red}$.

Let $J_0 := \langle x^{u+} - x^{u-} : u \in \mathcal{B}_{red}\rangle$.

for $i = 1, \ldots, n$ **do**

Compute $J_i = (J_{i-1} : x_i^{\infty})$ as:

(a) Compute $\mathcal{G}_{i-1}$ the reduced Gröbner basis for J_{i-1} with respect to $\prec_{\omega}^{gr_i}$.

(b) Divide each element $f \in \mathcal{G}_{i-1}$ by the highest power of x_i that divides f.

output: $J_n = \{x^{u_1} - x^{v_1}, \ldots, x^{u_s} - x^{v_s}\}$ a system of generators for I_A.

Algorithm 8: Pareto-optimal solutions computation for $MIP_{A,C}(b)$

input : $MIP_{A,C}(b)$

Step 1.: Compute an initial feasible solution, α_o, for $MIP_{A,C}(b)$.

Step 2.: Compute a system of generators for I_A: $\{x^{u_1} - x^{v_1}, \ldots, x^{u_s} - x^{v_s}\}$, using `setofgenerators(`$A$`)`.

Step 3.: Compute the partial reduced Gröbner basis for $I_A = \langle x^{u_1} - x^{v_1}, \ldots, x^{u_s} - x^{v_s}\rangle$, $G = \{G_1, \ldots, G_t\}$.

Step 4.: Calculate the set of partial remainders: $R := pR(x_o^{\alpha}, \mathcal{G}_C)$.

output: Pareto-optimal Solutions : R.

Example 2.3.1.

$$\begin{array}{lll} \min & \{10x + y, x + 10y\} & \\ s.a. & & \\ & 2x + 2y & \geqslant 17 \\ & y & \leqslant 11 \\ & x, y & \in \mathbb{Z}_+ \end{array} \tag{7}$$

Transforming the problem to the standard form:

$$\begin{array}{lll} \min & \{10x + y + 0t, x + 10y + 0t\} & \\ s.a. & & \\ & 2x + 2y - z & = 17 \\ & 2y + t & = 11 \\ & x, y, z, t & \in \mathbb{Z}_+ \end{array} \tag{8}$$

Following the steps of algorithms 7:

(1) *Basis for the* $Ker(A)$ *:* $\mathcal{B} := \{(1, -1, 0, 2), (1, 0, 2, 0)\}$.

(2) *LLL-reduced basis for* $\mathcal{B}$ *:* $\mathcal{B}_{red} := \mathcal{B} := \{(1,-1,0,2),(1,0,2,0)\}$.[1]

(3) $J_0 := \langle x^{u_+} - x^{u_-} : u \in \mathcal{B}_{red}\rangle = \langle x_1x_4^2 - x_2, x_1x3^2 - 1\rangle$

(4) $J_{i+1} := (J_i : x_i^\infty)$

(a) $\mathcal{G}_0 := \{x_1x_3^2 - 1, x_1x_4^2 - x_2\} \Rightarrow J_1 := \langle x_1x3^2 - 1, x_1x_4^2 - x_2\rangle$

(b) $\mathcal{G}_1 := \{x_1x_4^2 - x_2, x_4^2 - x_2x_3^2, x_1x_3^2 - 1\} \Rightarrow J_2 := \langle x_1x_4^2 - x_2, x_4^2 - x_2x_3^2, x_1x_3^2 - 1\rangle$

(c) $\mathcal{G}_2 := \{x_1x_3^2 - 1, x_4^2 - x_2x_3^2\} \Rightarrow J_3 := \langle x_1x_3^2 - 1, x_4^2 - x_2x_3^2\rangle$

(d) $\mathcal{G}_3 := \{x_4^2 - x_2x_3^2, x_1x_4^2 - x_2, x_1x_3^2 - 1\} \Rightarrow J_4 := \langle x_4^2 - x_2x_3^2, x_1x_4^2 - x_2, x_1x_3^2 - 1\rangle$

(5) *p-Gröbner basis for* $I_A = \langle x_4^2 - x_2x_3^2, x_1x_4^2 - x_2, x_1x_3^2 - 1\rangle$ *:*

$\mathcal{G}_1 := :\ \{(x_4^2 - x_2x_3^2, -x_2x_3^2), (x_1x_4^2 - x_2, -x_2), (x_3^2 - x_4^2, x_3^2)\}$

$\mathcal{G}_2 := :\ \{(x_1x_4^2 - x_2, x_1x_4^2), (x_1x_3^2 - 1, x_1x_3^2), (x_3^2 - x_4^2, x_3^2)\}$

(6) *Finding a feasible solution* [2] $u = (9,0,1,11)$ *is a feasible solution.*

(7) *Reduction of* $x^u = x_1^9x_3x_4^11$*, by* $\mathcal{G}$ *and by all permutations of* $(1,2)$*, i.e,* $\{(1,2),(2,1)\}$

(a) $PR(x_1^9x_3x_4^11, \mathcal{G}_1) = \{(x_1^9x_3x_4^11)\}$ *and*
$PR(x_1^9x_3x_4^11, \mathcal{G}_2) = \{x_1^9x_3x_4^11, x_1^8x_2x_3x_4^9, x_1^7x_2^2x_3x_4^7, x_1^6x_2^3x_3x_4^5, x_1^5x_2^4x_3x_4^3, x_1^4x_2^5x_3x_4\}$.

(b) $PR(x_1^9x_3x_4^11, \mathcal{G}_2) = \{x_1^9x_3x_4^11, x_1^8x_2x_3x_4^9, x_1^7x_2^2x_3x_4^7, x_1^6x_2^3x_3x_4^5, x_1^5x_2^4x_3x_4^3, x_1^4x_2^5x_3x_4\}$ *and:*

- $PR(x_1^9x_3x_4^11, \mathcal{G}_1) = \{x_1^9x_3x_4^11\}$
- $PR(x_1^8x_2x_3x_4^11, \mathcal{G}_1) = \{x_1^8x_2x_3x_4^11, x_1^9x_3x_4^11\}$
- $PR(x_1^7x_2^2x_3x_4^11, \mathcal{G}_1) = \{x_1^7x_2^2x_3x_4^11, x_1^8x_2x_3x_4^11, x_1^9x_3x_4^11\}$
- $PR(x_1^6x_2^3x_3x_4^{11}, \mathcal{G}_1) = \{x_1^6x_2^3x_3x_4^{11}, x_1^7x_2^2x_3x_4^{11}, x_1^8x_2x_3x_4^{11}, x_1^9x_3x_4^{11}\}$
- $PR(x_1^5x_2^4x_3x_4^{11}, \mathcal{G}_1) = \{x_1^5x_2^4x_3x_4^{11}, x_1^6x_2^3x_3x_4^{11}, x_1^7x_2^2x_3x_4^{11}, x_1^8x_2x_3x_4^{11}, x_1^9x_3x_4^{11}\}$
- $PR(x_1^4x_2^5x_3x_4^{11}, \mathcal{G}_1) = \{x_1^4x_2^5x_3x_4^{11}, x_1^5x_2^4x_3x_4^{11}, x_1^6x_2^3x_3x_4^{11}, x_1^7x_2^2x_3x_4^{11}, x_1^8x_2x_3x_4^{11}, x_1^9x_3x_4^{11}\}$

(c) *Pareto Optimal Solutions:*

$$\{(9,0,1,11),(8,1,1,9),(7,2,1,7),(6,3,1,5),(5,4,1,3),(4,5,1,1)\}$$

Figure 3.7 shows the feasible region and the Pareto-optimal solutions of the example above.

[1] In this case, the LLL-reduced basis for the kernel of A is the same as the initial computed basis.

[2] Mathematica allows to find instances for systems of diophantine linear equations with the function `FindInstance[]`

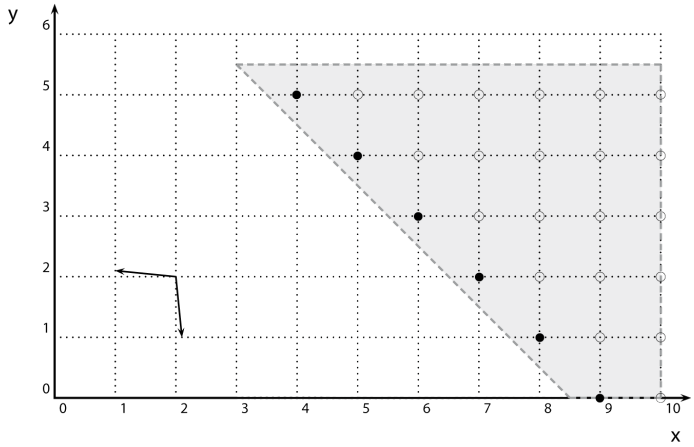


FIGURE 2.2. Feasible region, Pareto-optimal solutions and improvement cone for Example 2.3.1

The following example shows how to proceed to apply partial Gröbner bases to obtain Pareto-optimal solutions of multiobjective linear integer problems with negative coefficient in the objective matrix.

EXAMPLE 2.3.2. *The same feasible region with some negative entries in the objective matrix:*

$$
\begin{array}{lrl}
\min & \{10x - y, x - 10y\} & \\
s.a. & & \\
& 2x + 2y & \geqslant 17 \\
& y & \leqslant 11 \\
& x & \leqslant 10 \\
& x, y & \in \mathbb{Z}_+
\end{array}
\tag{9}
$$

We modify the problem to avoid inequalities and negative coefficients in the objective matrix. The last equation corresponds with the identification $w_2 = 10 + c_2\, x$. *The 10 is added to avoid negative values in the variable* w_2 *since values are treated as exponents of polynomials and negative exponents are not allowed.*

$$
\begin{array}{lrl}
\min & \{w_1, w_2\} & \\
s.a. & 2x + 2y - q & = 17 \\
& 2y + t & = 11 \\
& x + z & = 10 \\
& -10x + y + w_1 & = 0 \\
& -x + 10y + w_2 & = 10 \\
x, y \in \mathbb{Z}_+, & z, t, q, w_1, w_2 \in \mathbb{R} &
\end{array}
\tag{10}
$$

With the same procedure as above we obtain the following p-Gröbner basis $\mathcal{G} = \{\mathcal{G}_1, \mathcal{G}_2, \mathcal{G}_3, \mathcal{G}_4, \mathcal{G}_5, \mathcal{G}_6, \mathcal{G}_7\}$*:*

$\mathcal{G}_1 = \{(x_4{}^2x_6{}^{11}x_1 - x_2x_5x_7{}^{11},\ x_4{}^2x_6{}^{11}x_1),\ (x_1{}^2x_3{}^2x_4{}^2x_6{}^{21} - x_2x_5{}^2x_7{}^{12},\ x_1{}^2x_3{}^2x_4{}^2x_6{}^{21})\}$

$\mathcal{G}_2 = \{(x_4^2x_6 - x_2x_3{}^2x_7{}^{10}, x_4^2x_6),\ (x_1^2x_3{}^2x_4{}^2x_6{}^{21} - x_2x_5{}^2x_7{}^{12},\ x_1{}^2x_3{}^2x_4{}^2x_6{}^{21})\}$

$\mathcal{G}_3 = (x_1x_4{}^4x_6{}^{12} - x_2{}^2x_3{}^2x_5x_7{}^{21},\ x_1x_4{}^4x_6{}^{12}),\ (x_1{}^2x_3{}^2x_4{}^2x_6{}^{21} - x_2x_5{}^2x_7{}^{12},\ x_1{}^2x_3{}^2x_4{}^2x_6{}^{21})\}$

$\mathcal{G}_4 = (-x_5x_7 + x_1x_3{}^2x_6{}^{10},\ x_1x_3{}^2x_6{}^{10}),\ (x_1{}^2x_3{}^2x_4{}^2x_6{}^{21} - x_2x_5{}^2x_7{}^{12},\ x_1{}^2x_3{}^2x_4{}^2x_6{}^{21})\}$

$\mathcal{G}_5 = (x_2x_5x_7{}^{11} - x_4{}^2x_6{}^{11}x_1,\ x_2x_5x_7{}^{11}),\ (x_2{}^2x_3{}^2x_5x_7{}^{21} - x_1x_4{}^4x_6{}^{12},\ x_2{}^2x_3{}^2x_5x_7{}^{21}),\ (x_2x_5{}^2x_7{}^{12} - x_1{}^2x_3{}^2x_4{}^2x_6{}^{21},\ x_2x_5{}^2x_7{}^{12})\}$

$\mathcal{G}_6 = (x_2{}^2x_3{}^2x_5x_7{}^{21} - x_1x_4{}^4x_6{}^{12},\ x_2{}^2x_3{}^2x_5x_7{}^{21}),\ (x_2x_5{}^2x_7{}^{12} - x_1{}^2x_3{}^2x_4{}^2x_6{}^{21},\ x_2x_5{}^2x_7{}^{12}),\ (-x_4{}^2x_6 + x_2x_3{}^2x_7{}^{10},\ x_2x_3{}^2x_7{}^{10})\}$

$\mathcal{G}_7 = (x_2{}^2x_3{}^2x_5x_7{}^{21} - x_1x_4{}^4x_6{}^{12},\ x_2{}^2x_3{}^2x_5x_7{}^{21}),\ (x_2x_5{}^2x_7{}^{12} - x_1{}^2x_3{}^2x_4{}^2x_6{}^{21},\ x_2x_5{}^2x_7{}^{12}),\ (x_5x_7 - x_1x_3{}^2x_6{}^{10},\ x_5x_7)\}$

From the initial solution $(x, y, z, t, q, w_1, w_2) = (9, 4, 9, 3, 1, 86, 31)$ *and computing partial reductions:*

- $pR(x_1^9\,x_2^4\,x_3^9\,x_4^3\,x_5\,x_6^{86}\,x_7^{41}, \mathcal{G}_1) = \{x_1^9\,x_2^4\,x_3^9\,x_4^3\,x_5\,x_6^{86}\,x_7^{41},\ x_1^{10}\,x_2^2\,x_3^9\,x_4\,x_5^2\,x_6^{75}\,x_7^{52}\}$
- $pR(x_1^8\,x_2^5\,x_3^9\,x_4\,x_5^2\,x_6^{75}\,x_7^{52}, \mathcal{G}_2) = \{x_1^8\,x_2^5\,x_3^9\,x_4\,x_5^2\,x_6^{75}\,x_7^{52}\}$
- $pR(x_1^8\,x_2^5\,x_3^9\,x_4\,x_5^2\,x_6^{75}\,x_7^{52}, \mathcal{G}_3) = \{x_1^8\,x_2^5\,x_3^9\,x_4\,x_5^2\,x_6^{75}\,x_7^{52}\}$
- $pR(x_1^8\,x_2^5\,x_3^9\,x_4\,x_5^2\,x_6^{75}\,x_7^{52}, \mathcal{G}_4) = \{x_1^8\,x_2^5\,x_3^9\,x_4\,x_5^2\,x_6^{75}\,x_7^{52},\ x_1^7\,x_2^5\,x_3^7\,x_4\,x_5^3\,x_6^{65}\,x_7^{53},\ x_1^6\,x_2^5\,x_3^5\,x_4\,x_5^4\,x_6^{55}\,x_7^{54},\ x_1^5\,x_2^5\,x_3^3\,x_4\,x_5^5\,x_6^{45}\,x_7^{55},\ x_1^4\,x_2^5\,x_3\,x_4\,x_5^6\,x_6^{35}\,x_7^{56}\}$
- $pR(x_1^8\,x_2^5\,x_3^9\,x_4\,x_5^2\,x_6^{75}\,x_7^{52}, \mathcal{G}_5) = \{x_1^8\,x_2^5\,x_3^9\,x_4\,x_5^2\,x_6^{75}\,x_7^{52},\ x_1^9\,x_2^4\,x_3^9\,x_4^3\,x_5\,x_6^{86}\,x_7^{41},\ x_1^{10}\,x_2^3\,x_3^9\,x_4^5\,x_6^{97}\,x_7^{30}\}$
  $pR(x_1^7\,x_2^5\,x_3^7\,x_4\,x_5^3\,x_6^{65}\,x_7^{53}, \mathcal{G}_5) = \{x_1^7\,x_2^5\,x_3^7\,x_4\,x_5^3\,x_6^{65}\,x_7^{53},\ x_1^8\,x_2^4\,x_3^7\,x_4^3\,x_5^2\,x_6^{76}\,x_7^{42},\ x_1^9\,x_2^3\,x_3^7\,x_4^5\,x_5\,x_6^{87}\,x_7^{31},\ x_1^{10}\,x_2^2\,x_3^7\,x_4^7\,x_6^{98}\,x_7^{20}\},$
  $pR(x_1^6\,x_2^5\,x_3^5\,x_4\,x_5^4\,x_6^{55}\,x_7^{54}, \mathcal{G}_5) = \{x_1^6\,x_2^5\,x_3^5\,x_4\,x_5^4\,x_6^{55}\,x_7^{54},\ x_1^7\,x_2^4\,x_3^5\,x_4^3\,x_5^3\,x_6^{66}\,x_7^{43},\ x_1^8\,x_2^3\,x_3^5\,x_4^5\,x_5^2\,x_6^{77}\,x_7^{32},\ x_1^9\,x_2^2\,x_3^5\,x_4^7\,x_5\,x_6^{88}\,x_7^{21},\ x_1^{10}\,x_3^3\,x_4^{11}\,x_6^{100}\}$
  $pR(x_1^5\,x_2^5\,x_3^3\,x_4\,x_5^5\,x_6^{45}\,x_7^{55}, \mathcal{G}_5) = \{x_1^5\,x_2^5\,x_3^3\,x_4\,x_5^5\,x_6^{45}\,x_7^{55},\ x_1^6\,x_2^4\,x_3^3\,x_4^3\,x_5^4\,x_6^{56}\,x_7^{44},\ x_1^7\,x_2^3\,x_3^5\,x_4^3\,x_5^5\,x_6^{67}\,x_7^{33},\ x_1^8\,x_2^2\,x_3^3\,x_4^7\,x_5^2\,x_6^{78}\,x_7^{12},\ x_1^9\,x_2\,x_3^3\,x_4^9\,x_5\,x_6^{89}\,x_7^{11},\ x_1^{10}\,x_3^3\,x_4^{11}\,x_6^{100}\}$
  $pR(x_1^4\,x_2^5\,x_3\,x_4\,x_5^6\,x_6^{35}\,x_7^{56}, \mathcal{G}_5) = \{x_1^4\,x_2^5\,x_3\,x_4\,x_5^6\,x_6^{35}\,x_7^{56},\ x_1^5\,x_2^4\,x_3\,x_4^3\,x_5^5\,x_6^{46}\,x_7^{45},\ x_1^6\,x_2^3\,x_3\,x_4^5\,x_5^4\,x_6^{57}\,x_7^{34},\ x_1^7\,x_2^2\,x_3\,x_4^7\,x_5^3\,x_6^{68}\,x_7^{23},\ x_1^8\,x_2\,x_3\,x_4^9\,x_5^2\,x_6^{79}\,x_7^{12},\ x_1^9\,x_2\,x_3^3\,x_4^9\,x_5\,x_6^{89}\,x_7^{11},\ x_1^9\,x_3\,x_4^{11}\,x_5\,x_6^{90}\,x_7\}$
- $pR(x_1^8\,x_2^5\,x_3^9\,x_4\,x_5^2\,x_6^{75}\,x_7^{52}, \mathcal{G}_6) = \{x_1^8\,x_2^5\,x_3^9\,x_4\,x_5^2\,x_6^{75}\,x_7^{52},\ x_1^9\,x_2^3\,x_3^7\,x_4^5\,x_5\,x_6^{87}\,x_7^{31},\ x_1^{10}\,x_2\,x_3^5\,x_4^9\,x_6^{99}\,x_7^{10},\ x_1^{10}\,x_3^3\,x_4^{11}\,x_6^{100}\}$
  $pR(x_1^5\,x_2^5\,x_3^3\,x_4\,x_5^5\,x_6^{45}\,x_7^{55}, \mathcal{G}_6) = \{x_1^5\,x_2^5\,x_3^3\,x_4\,x_5^5\,x_6^{45}\,x_7^{55},\ x_1^6\,x_2^3\,x_3\,x_4^5\,x_5^4\,x_6^{57}\,x_7^{34},\ x_1^{10}\,x_2\,x_3^5\,x_4^9\,x_6^{99}\,x_7^{10},\ x_1^8\,x_2^2\,x_3^3\,x_4^7\,x_5^2\,x_6^{78}\,x_7^{22},\ x_1^{10}\,x_3^3\,x_4^{11}\,x_6^{100}\}$
  $pR(x_1^4\,x_2^5\,x_3\,x_4\,x_5^6\,x_6^{35}\,x_7^{56}, \mathcal{G}_6) = \{x_1^{10}\,x_2\,x_3^5\,x_4^9\,x_6^{99}\,x_7^{10},\ x_1^{10}\,x_2^2\,x_3^7\,x_4^7\,x_6^{98}\,x_7^{20},$

$x_1^8\,x_2^3\,x_3^5\,x_4^5\,x_5^2\,x_6^{77}\,x_7^{32},\ x_1^6\,x_2^4\,x_3^3\,x_4^3\,x_5^4\,x_6^{56}\,x_7^{44},\ x_1^4\,x_2^5\,x_3\,x_4\,x_5^6\,x_6^{35}\,x_7^{56},$
$x_1^{10}\,x_3^3\,x_4^{11}\,x_6^{100}\}$
$pR(x_1^9\,x_2^4\,x_3^9\,x_4^3\,x_5\,x_6^{86}\,x_7^{41},\mathcal{G}_6) = \{x_1^9\,x_2^4\,x_3^9\,x_4^3\,x_5\,x_6^{86}\,x_7^{41},\ x_1^{10}\,x_2\,x_3^5\,x_4^9\,x_6^{99}\,x_7^{10},$
$x_1^{10}\,x_2^2\,x_3^7\,x_4^7\,x_6^{98}\,x_7^{20},\ x_1^{10}\,x_3^3\,x_4^{11}\,x_6^{100}\}$
$pR(x_1^{10}\,x_2^3\,x_3^9\,x_4^5\,x_6^{97}\,x_7^{30},\mathcal{G}_6) = \{x_1^{10}\,x_2\,x_3^5\,x_4^9\,x_6^{99}\,x_7^{10},\ x_1^{10}\,x_2^2\,x_3^7\,x_4^7\,x_6^{98}\,x_7^{20},$
$x_1^{10}\,x_2^3\,x_3^9\,x_4^5\,x_6^{97}\,x_7^{30},\ x_1^{10}\,x_3^3\,x_4^{11}\,x_6^{100}\}$
$pR(x_1^7\,x_2^5\,x_3^7\,x_4\,x_5^3\,x_6^{65}\,x_7^{53},\mathcal{G}_6) = \{x_1^9\,x_2\,x_3^3\,x_4^9\,x_5\,x_6^{89}\,x_7^{11},\ x_1^8\,x_2^3\,x_3^5\,x_4^5\,x_5^2\,x_6^{77}\,x_7^{32},$
$x_1^9\,x_3\,x_4^{11}\,x_5\,x_6^{90}\,x_7,\ x_1^7\,x_2^5\,x_3^7\,x_4\,x_5^3\,x_6^{65}\,x_7^{53}\}$
$pR(x_1^8\,x_2^4\,x_3^7\,x_4^3\,x_5^2\,x_6^{76}\,x_7^{42},\mathcal{G}_6) = \{x_1^9\,x_2^2\,x_3^5\,x_4^7\,x_5\,x_6^{88}\,x_7^{21},\ x_1^8\,x_2^4\,x_3^7\,x_4^3\,x_5^2\,x_6^{76}\,x_7^{42},$
$x_1^{10}\,x_3^3\,x_4^{11}\,x_6^{100}\}$
$pR(x_1^9\,x_2^3\,x_3^7\,x_4^5\,x_5\,x_6^{87}\,x_7^{31},\mathcal{G}_6) = \{x_1^9\,x_2^3\,x_3^7\,x_4^5\,x_5\,x_6^{87}\,x_7^{31},\ x_1^{10}\,x_2\,x_3^5\,x_4^9\,x_6^{99}\,x_7^{10},$
$x_1^{10}\,x_3^3\,x_4^{11}\,x_6^{100}\}$
$pR(x_1^{10}\,x_2^2\,x_3^7\,x_4^7\,x_6^{98}\,x_7^{20},\mathcal{G}_6) = \{x_1^{10}\,x_2\,x_3^5\,x_4^9\,x_6^{99}\,x_7^{10},\ x_1^{10}\,x_2^2\,x_3^7\,x_4^7\,x_6^{98}\,x_7^{20},$
$x_1^{10}\,x_3^3\,x_4^{11}\,x_6^{100}\}$
$pR(x_1^6\,x_2^5\,x_3^5\,x_4\,x_5^4\,x_6^{55}\,x_7^{54},\mathcal{G}_6) = \{x_1^8\,x_2\,x_3\,x_4^9\,x_5^2\,x_6^{79}\,x_7^{12},\ x_1^6\,x_2^5\,x_3^5\,x_4\,x_5^4\,x_6^{55}\,x_7^{54},$
$x_1^{10}\,x_3^3\,x_4^{11}\,x_6^{100},$
$x_1^7\,x_2^3\,x_3^3\,x_4^5\,x_5^3\,x_6^{67}\,x_7^{33}\}$
$pR(x_1^7\,x_2^4\,x_3^5\,x_4^3\,x_5^3\,x_6^{66}\,x_7^{43},\mathcal{G}_6) = \{x_1^7\,x_2^4\,x_3^5\,x_4^3\,x_5^3\,x_6^{66}\,x_7^{43},\ x_1^8\,x_2^2\,x_3^3\,x_4^7\,x_5^2\,x_6^{78}\,x_7^{22},$
$x_1^9\,x_3\,x_4^{11}\,x_5\,x_6^{90}\,x_7\}$
$pR(x_1^8\,x_2^3\,x_3^5\,x_4^5\,x_5^2\,x_6^{77}\,x_7^{32},\mathcal{G}_6) = \{x_1^9\,x_2\,x_3^3\,x_4^9\,x_5\,x_6^{89}\,x_7^{11},\ x_1^8\,x_2^3\,x_3^5\,x_4^5\,x_5^2\,x_6^{77}\,x_7^{32},$
$x_1^9\,x_3\,x_4^{11}\,x_5\,x_6^{90}\,x_7\}$
$pR(x_1^9\,x_2^2\,x_3^5\,x_4^7\,x_5\,x_6^{88}\,x_7^{21},\mathcal{G}_6) = \{x_1^9\,x_2^2\,x_3^5\,x_4^7\,x_5\,x_6^{88}\,x_7^{21},\ x_1^{10}\,x_3^3\,x_4^{11}\,x_6^{100}\}$
$pR(x_1^{10}\,x_3^3\,x_4^{11}\,x_6^{100},\mathcal{G}_6) = \{x_1^{10}\,x_3^3\,x_4^{11}\,x_6^{100}\}$
$pR(x_1^6\,x_2^4\,x_3^3\,x_4^3\,x_5^4\,x_6^{56}\,x_7^{44},\mathcal{G}_6) = \{x_1^9\,x_2\,x_3^3\,x_4^9\,x_5\,x_6^{89}\,x_7^{11},\ x_1^6\,x_2^4\,x_3^3\,x_4^3\,x_5^4\,x_6^{56}\,x_7^{44},$
$x_1^7\,x_2^2\,x_3\,x_4^7\,x_5^3\,x_6^{68}\,x_7^{23},\ x_1^9\,x_3\,x_4^{11}\,x_5\,x_6^{90}\,x_7\}$
$pR(x_1^7\,x_2^3\,x_3^3\,x_4^5\,x_5^3\,x_6^{67}\,x_7^{33},\mathcal{G}_6) = \{x_1^8\,x_2\,x_3\,x_4^9\,x_5^2\,x_6^{79}\,x_7^{12},\ x_1^{10}\,x_3^3\,x_4^{11}\,x_6^{100},$
$x_1^7\,x_2^3\,x_3^3\,x_4^5\,x_5^3\,x_6^{67}\,x_7^{33}\}$
$pR(x_1^8\,x_2^2\,x_3^3\,x_4^7\,x_5^2\,x_6^{78}\,x_7^{12},\mathcal{G}_6) = \{x_1^{10}\,x_2\,x_3^5\,x_4^9\,x_6^{99},\ x_1^8\,x_2^2\,x_3^3\,x_4^7\,x_5^2\,x_6^{78}\,x_7^{12}\}$
$pR(x_1^9\,x_2\,x_3^3\,x_4^9\,x_5\,x_6^{89}\,x_7^{11},\mathcal{G}_6) = \{x_1^9\,x_2\,x_3^3\,x_4^9\,x_5\,x_6^{89}\,x_7^{11},\ x_1^9\,x_3\,x_4^{11}\,x_5\,x_6^{90}\,x_7\}$
$pR(x_1^5\,x_2^4\,x_3\,x_4^3\,x_5^5\,x_6^{46}\,x_7^{45},\mathcal{G}_6) = \{x_1^5\,x_2^4\,x_3\,x_4^3\,x_5^5\,x_6^{46}\,x_7^{45},\ x_1^9\,x_2\,x_3^3\,x_4^9\,x_5\,x_6^{89}\,x_7^{11},$
$x_1^9\,x_2^2\,x_3^5\,x_4^7\,x_5\,x_6^{88}\,x_7^{21},\ x_1^9\,x_3\,x_4^{11}\,x_5\,x_6^{90}\,x_7,\ x_1^7\,x_2^3\,x_3^3\,x_4^5\,x_5^3\,x_6^{67}\,x_7^{33}\}$
$pR(x_1^6\,x_2^3\,x_3\,x_4^5\,x_5^4\,x_6^{57}\,x_7^{34},\mathcal{G}_6) = \{x_1^6\,x_2^3\,x_3\,x_4^5\,x_5^4\,x_6^{57}\,x_7^{34},\ x_1^{10}\,x_2\,x_3^5\,x_4^9\,x_6^{99}\,x_7^{10},$
$x_1^8\,x_2^2\,x_3^3\,x_4^7\,x_5^2\,x_6^{78}\,x_7^{22},\ x_1^{10}\,x_3^3\,x_4^{11}\,x_6^{100}\}$
$pR(x_1^7\,x_2^2\,x_3\,x_4^7\,x_5^3\,x_6^{68}\,x_7^{33},\mathcal{G}_6) = \{x_1^9\,x_3\,x_4^{11}\,x_5\,x_6^{90}\,x_7^{11},\ x_1^7\,x_2^2\,x_3\,x_4^7\,x_5^3\,x_6^{68}\,x_7^{33},$
$x_1^9\,x_2\,x_3^3\,x_4^9\,x_5\,x_6^{89}\,x_7^{21}\}$
$pR(x_1^8\,x_2\,x_3\,x_4^9\,x_5^2\,x_6^{79}\,x_7^{12},\mathcal{G}_6) = \{x_1^8\,x_2\,x_3\,x_4^9\,x_5^2\,x_6^{79}\,x_7^{12},\ x_1^{10}\,x_3^3\,x_4^{11}\,x_6^{100}\}$
$pR(x_1^9\,x_2\,x_3^3\,x_4^9\,x_5\,x_6^{89}\,x_7^{11},\mathcal{G}_6) = \{x_1^9\,x_2\,x_3^3\,x_4^9\,x_5\,x_6^{89}\,x_7^{11},\ x_1^9\,x_3\,x_4^{11}\,x_5\,x_6^{90}\,x_7\}$
$pR(x_1^9\,x_3\,x_4^{11}\,x_5\,x_6^{90}\,x_7,\mathcal{G}_6) = \{x_1^9\,x_3\,x_4^{11}\,x_5\,x_6^{90}\,x_7\}$

- $pR(x_1^5\,x_2^5\,x_3^3\,x_4\,x_5^5\,x_6^{45}\,x_7^{55},\mathcal{G}_7) = \{x_1^{10}\,x_2\,x_3^5\,x_4^9\,x_6^{99}\,x_7^{10},\ x_1^5\,x_2^5\,x_3^3\,x_4\,x_5^5\,x_6^{45}\,x_7^{55},$
$x_1^8\,x_2^2\,x_3^3\,x_4^7\,x_5^2\,x_6^{78}\,x_7^{22},\ x_1^6\,x_2^3\,x_3\,x_4^5\,x_5^4\,x_6^{57}\,x_7^{34}\}$
$pR(x_1^4\,x_2^5\,x_3\,x_4\,x_5^6\,x_6^{35}\,x_7^{56},\mathcal{G}_7) = \{x_1^4\,x_2^5\,x_3\,x_4\,x_5^6\,x_6^{35}\,x_7^{56},\ x_1^{10}\,x_2^2\,x_3^7\,x_4^7\,x_6^{98}\,x_7^{20},$
$x_1^8\,x_2^3\,x_3^5\,x_4^5\,x_5^2\,x_6^{77}\,x_7^{32},\ x_1^6\,x_2^4\,x_3^3\,x_4^3\,x_5^4\,x_6^{56}\,x_7^{44}\}$

$pR(x_1^{10}\,x_3^3\,x_4^{11}\,x_6^{100},\,\mathcal{G}_7) = \{x_1^{10}\,x_3^3\,x_4^{11}\,x_6^{100}\}$
$pR(x_1^6\,x_2^4\,x_3^3\,x_4^3\,x_5^4\,x_6^{56}\,x_7^{44},\,\mathcal{G}_7) = \{x_1^7\,x_2^2\,x_3\,x_4^7\,x_5^3\,x_6^{68}\,x_7^{23},\,x_1^{10}\,x_2\,x_3^5\,x_4^9\,x_6^{99}\,x_7^{10},$
$x_1^6\,x_2^4\,x_3^3\,x_4^3\,x_5^4\,x_6^{56}\,x_7^{44},\,x_1^9\,x_2\,x_3^3\,x_4^9\,x_5\,x_6^{89}\,x_7^{11}\}$
$pR(x_1^7\,x_2^3\,x_3^3\,x_4^5\,x_5^3\,x_6^{67}\,x_7^{33},\,\mathcal{G}_7) = \{x_1^8\,x_2\,x_3\,x_4^9\,x_5^2\,x_6^{79}\,x_7^{12},\,x_1^7\,x_2^3\,x_3^3\,x_4^5\,x_5^3\,x_6^{67}\,x_7^{33},$
$x_1^{10}\,x_3^3\,x_4^{11}\,x_6^{100}\}$
$pR(x_1^8\,x_2^2\,x_3^3\,x_4^7\,x_5^2\,x_6^{78}\,x_7^{22},\,\mathcal{G}_7) = \{x_1^{10}\,x_3^3\,x_4^{11}\,x_6^{100},\,x_1^8\,x_2^2\,x_3^3\,x_4^7\,x_5^2\,x_6^{78}\,x_7^{22},$
$x_1^9\,x_3\,x_4^{11}\,x_5\,x_6^{90}\,x_7\}$
$pR(x_1^9\,x_2\,x_3^3\,x_4^9\,x_5\,x_6^{89}\,x_7^{11},\,\mathcal{G}_7) = \{x_1^{10}\,x_2\,x_3^5\,x_4^9\,x_6^{99}\,x_7^{10},\,x_1^9\,x_2\,x_3^3\,x_4^9\,x_5\,x_6^{89}\,x_7^{11}\}$
$pR(x_1^5\,x_2^4\,x_3\,x_4^3\,x_5^5\,x_6^{46}\,x_7^{45},\,\mathcal{G}_7) = \{x_1^7\,x_2^3\,x_3^3\,x_4^5\,x_5^3\,x_6^{67}\,x_7^{33},\,x_1^{10}\,x_2^2\,x_3^7\,x_4^7\,x_6^{98}\,x_7^{20},$
$x_1^9\,x_2^2\,x_3^5\,x_4^7\,x_5\,x_6^{88}\,x_7^{21},\,x_1^5\,x_2^4\,x_3\,x_4^3\,x_5^5\,x_6^{46}\,x_7^{45}\}$
$pR(x_1^6\,x_2^3\,x_3\,x_4^5\,x_5^4\,x_6^{57}\,x_7^{34},\,\mathcal{G}_7) = \{x_1^{10}\,x_2\,x_3^5\,x_4^9\,x_6^{99}\,x_7^{10},\,x_1^8\,x_2^2\,x_3^3\,x_4^7\,x_5^2\,x_6^{78}\,x_7^{22},$
$x_1^6\,x_2^3\,x_3\,x_4^5\,x_5^4\,x_6^{57}\,x_7^{34}\}$
$pR(x_1^7\,x_2^2\,x_3\,x_4^7\,x_5^3\,x_6^{68}\,x_7^{23},\,\mathcal{G}_7) = \{x_1^7\,x_2^2\,x_3\,x_4^7\,x_5^3\,x_6^{68}\,x_7^{23},\,x_1^{10}\,x_2\,x_3^5\,x_4^9\,x_6^{99}\,x_7^{10},$
$x_1^9\,x_2\,x_3^3\,x_4^9\,x_5\,x_6^{89}\,x_7^{11}\}$
$pR(x_1^8\,x_2\,x_3\,x_4^9\,x_5^2\,x_6^{79}\,x_7^{12},\,\mathcal{G}_7) = \{x_1^8\,x_2\,x_3\,x_4^9\,x_5^2\,x_6^{79}\,x_7^{12},\,x_1^{10}\,x_3^3\,x_4^{11}\,x_6^{100}\}$
$pR(x_1^9\,x_3\,x_4^{11}\,x_5\,x_6^{90}\,x_7,\,\mathcal{G}_7) = \{x_1^{10}\,x_3^3\,x_4^{11}\,x_6^{100},\,x_1^9\,x_3\,x_4^{11}\,x_5\,x_6^{90}\,x_7\}.$

Then $pR(x_1^9\,x_2^4\,x_3^9\,x_4^3\,x_5\,x_6^{86}\,x_7^{41}, \mathcal{G}) = \{x_1^8\,x_2\,x_3\,x_4^9\,x_5^2\,x_6^{79}\,x_7^{12}$
$, x_1^7\,x_2^3\,x_3^3\,x_4^5\,x_5^3\,x_6^{67}\,x_7^{33},\,x_1^7\,x_2^2\,x_3\,x_4^7\,x_5^3\,x_6^{68}\,x_7^{23},\,x_1^{10}\,x_3^3\,x_4^{11}\,x_6^{100},$
$x_1^5\,x_2^5\,x_3^3\,x_4\,x_5^5\,x_6^{45}\,x_7^{55},\,x_1^8\,x_2^2\,x_3^3\,x_4^7\,x_5^2\,x_6^{78}\,x_7^{22},\,x_1^6\,x_2^3\,x_3\,x_4^5\,x_5^4\,x_6^{57}\,x_7^{34},$
$x_1^4\,x_2^5\,x_3\,x_4\,x_5^6\,x_6^{35}\,x_7^{56},\,x_1^6\,x_2^4\,x_3^3\,x_4^3\,x_5^4\,x_6^{56}\,x_7^{44},\,x_1^9\,x_2\,x_3^3\,x_4^9\,x_5\,x_6^{89}\,x_7^{11},$
$x_1^9\,x_3\,x_4^{11}\,x_5\,x_6^{90}\,x_7,\,x_1^5\,x_2^4\,x_3\,x_4^3\,x_5^5\,x_6^{46}\,x_7^{45}\}$

Hence, the entire set of Pareto Optimal Solutions (in the first two variables, the original ones) is:

$$\{(5,5),(4,5),(10,0),(6,4),(7,3),(8,2),(9,1),(5,4),(6,3),(7,2),(8,1),(9,0)\}$$

Figure 2.3 shows the feasible region and the Pareto-optimal solutions of the example above.

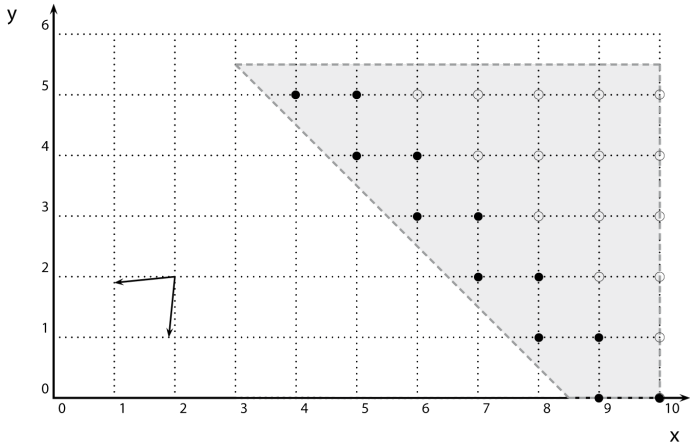


FIGURE 2.3. Feasible region, Pareto-optimal solutions and improvement cone for Example 2.3.2

CHAPTER 3

Partial Gröbner bases: a geometric approach

This chapter describes the geometrical interpretation of the results given in Chapter 2 for solving multiobjective linear integer problems. In that chapter we presented a general theory for partial Gröbner bases and then, we applied it to solve MOILP. For that application the results were applied to a special kind of polynomial ideals: toric ideals. Here, an analogous theory is described in a geometrical language and some of the results given for polynomials are geometrically interpreted. The p-Gröbner bases for toric ideals can be seen as geometric structures, a *geometric p-Gröbner basis* will be the vector translation of that notion. The main property of a geometric p-Gröbner basis being that for each pair in $\mathbb{Z}^n \times \mathbb{Z}^n_+$ with first component in $Ker(A)$, the reduction by maximal chains in the basis is the zero set. Then, we introduce the notion of test family for a multiobjective integer linear problem where the right-hand side varies (with objective and contraint matrices fixed). Then, the relationship between the geometrical interpretation of a p-Gröbner basis and test families is proven.

To show the analogy between the notions and the results of the algebraic and the geometric approaches, the same examples analyzed in Chapter 2 are presented here in terms of vectors rather than polynomials.

We propose, as in Chapter 2, two versions of the same algorithm to solve multiobjective integer programs based on this new construction: those based on the Conti-Traverso and the Hoşten-Sturmfels methods.

Both algorithms have been implemented and we report on some computational experiments based on the first version of the algorithm and on two different families of problems with different number of objective functions. The final part of the chapter is devoted to the results of the computational experiments and its analysis. Here, we solve several families of MOILP, report on the performance of the algorithms and draw some conclusions on its results and their implications.

3.1. From polynomials to vectors

In this chapter we are also interested in solving multiobjective linear integer problems (MOLIP) in the form:

$$\begin{aligned} \min \quad & (c_1\, x, \ldots, c_k\, x) \\ s.t. \quad & A\, x = b \\ & x \in \mathbb{Z}_+^n \end{aligned}$$

with $A \in \mathbb{Z}^{m\times n}$, $b \in \mathbb{Z}_+^m$ and $C = (c_i) \in \mathbb{Z}_+^{k\times n}$.

Our matrix A is encoded in the set

$$J_A = \{\{u, v\} : u, v \in \mathbb{N}^n, u - v \in Ker(A)\}. \tag{11}$$

Let $\pi : \mathbb{N}^n \longrightarrow \mathbb{Z}^n$ denote the map $x \mapsto Ax$. Given a right-hand side vector b in $\mathbb{Z}^n$, the set of feasible solutions to $MIP_{A,C}(b)$ constitutes $\pi^{-1}(b)$, the preimage of b under this map. In the rest of the chapter, we identify the discrete set of points $\pi^{-1}(b)$ with its convex hull and we call it the b-fiber of $MIP_{A,C}$. Thus, $\pi^{-1}(b)$ or the b-fiber of $MIP_{A,C}$ is the polyhedron defined by the convex hull of all feasible solutions to $MIP_{A,C}(b)$. First, we fix a partially admissible ordering over $\mathbb{N}$ induced by the objective function as follows:

$$u \prec_C v :\Longleftrightarrow C\, u \lneqq C\, v.$$

Then, for any pair $\{u, v\}$, with $u, v \in \mathbb{N}^n$, we define the set $setld(u, v)$ as follows:

$$setld(u,v) = \begin{cases} \{u\} & \text{if } v \prec_C u \\ \{v\} & \text{if } u \prec_C v \\ \{u,v\} & \text{if } u \text{ and } v \text{ are incomparable by } \prec_C \end{cases}$$

The reader may note that $setld(u, v)$ is the set of degrees of the leading monomials according to the identification $\{u, v\} \mapsto x^u - x^v \in \mathbb{R}[x_1, \ldots, x_n]$, induced by the partial order $\prec_C$.

From the above definition, $setld(u, v)$ may have more than one leading term, since $\prec_C$ is only a partial order. To account for all this information we denote by $\Upsilon(u, v)$ the set of triplets

$$\Upsilon(u, v) = \{(u, v, w) : w \in setld(u, v)\}.$$

With the identification between binomials and pairs described above, $\Upsilon(u, v)$ is partially identified with $\mathcal{F}(x^u - x^v)$.As for the map $\mathcal{F}$ in Chapter 2, the above concept extends to any finite set of pairs of vectors in $\mathbb{N}^n$, accordingly. For a pair of sets $\mathbf{u} = \{u_1, \ldots, u_t\}$ and $\mathbf{v} = \{v_1, \ldots, v_t\}$ the corresponding set

of ordered pairs is:

$$\Upsilon(\mathbf{u},\mathbf{v}) = \{(u_i, v_i, w) : w \in setld(u_i, v_i), i = 1, \ldots, t\}.$$

$\Upsilon(\mathbf{u},\mathbf{v})$ can be partially ordered based on the third component of its elements. Therefore, we can see $\Upsilon(\mathbf{u},\mathbf{v})$ as a directed graph $G(E,V)$ where V is identified with the elements of $\Upsilon(\mathbf{u},\mathbf{v})$ and $((u_i, v_i, w^{'}), (u_j, v_j, w)) \in E$ if $(u_i, v_i, w), (u_j, v_j, w^{'}) \in V$ and $w^{'} \prec_C w$. We are interested in the maximal ordered chains of G. Note that they can be efficiently computed by different methods, e.g. [5], [93].
The above concepts are clarified in the following example that is the same (but in terms of vectors) that Example 2.1.1.

EXAMPLE 3.1.1. *Let* $\mathbf{u} = \{(2,3), (0,2), (3,0), (2,1), (1,1)\}$, $\mathbf{v} = \{(1,4), (1,3), (4,2), (1,2), (1,0)\}$ *and* $\prec_C$ *the partial order induced by the matrix*

$$C = \begin{bmatrix} 2 & 1 \\ 3 & 5 \end{bmatrix}$$

then, $setld((2,3),(1,4)) = \{(2,3),(1,4)\}$, $setld((0,2),(1,3)) = \{(1,3)\}$, $setld((3,0),(4,2)) = \{(4,2)\}$, $setld((2,1),(1,2)) = \{(2,1),(1,2)\}$ *and* $setld((1,1),(1,0)) = \{(1,1)\}$. *Now, by definition we have:*

$$\begin{aligned}\Upsilon(\mathbf{u},\mathbf{v}) = \{ & \big((2,3),(1,4),(2,3)\big), \big((2,3),(1,4),(1,4)\big), \quad \big((0,2),(1,3),(1,3)\big), \\ & \big((3,0),(4,2),(4,2)\big), \big((2,1),(1,2),(2,1)\big), \quad \big((2,1),(1,2),(1,2)\big), \\ & \big((1,1),(1,0),(1,1)\big)\}.\end{aligned}$$

Figure 3.1 corresponds to the directed graph associated with $\Upsilon(\mathbf{u},\mathbf{v})$, *according to the partial ordering induced by* C. *There are four maximal chains:*
$M_1 = \{\big((3,0),\ (4,2),\ (4,2)\big),\ \big((2,3),\ (1,4),\ (2,3)\big),\ \big((0,2),\ (1,3),\ (1,3)\big),$
$\big((2,1),\ (1,2),\ (2,1)\big),\ \big((1,1),\ (1,0),\ (1,1)\big)\}$,
$M_2 = \{\big((3,0),\ (4,2),\ (4,2)\big),\ \big((2,3),\ (1,4),\ (2,3)\big),\ \big((0,2),\ (1,3),\ (1,3)\big),$
$\big((2,1),\ (1,2),\ (1,2)\big),\ \big((1,1),\ (1,0),\ (1,1)\big)\}$,
$M_3 = \{\big((2,3),\ (1,4),\ (1,4)\big),\ \big((0,2),\ (1,3),\ (1,3)\big),\ \big((2,1),\ (1,2),\ (2,1)\big),$
$\big((1,1),\ (1,0),\ (1,1)\big)\}$,
$M_4 = \{\big((2,3),\ (1,4),\ (1,4)\big),\ \big((0,2),\ (1,3),\ (1,3)\big),\ \big((2,1),\ (1,2),\ (1,2)\big),$
$\big((1,1),\ (1,0),\ (1,1)\big)\}$.

For any pair of sets $\mathbf{u} = \{u_1, \ldots, u_t\}$ and $\mathbf{v} = \{v_1, \ldots, v_t\}$ with $\{u_i, v_i\} \in J_A$, for all $i = 1, \ldots, t$, the corresponding set $\Upsilon(\mathbf{u},\mathbf{v})$ may also be seen as a set of pairs in $\mathbb{Z}^n \times \mathbb{Z}^n_+$ through the following map

$$\begin{aligned}\phi\colon \mathbb{N}^n \times \mathbb{N}^n \times \mathbb{N}^n \quad &\longrightarrow \mathbb{Z}^n \times \mathbb{Z}^n_+ \\ (u, v, w) \quad &\mapsto (u - v, w).\end{aligned}$$

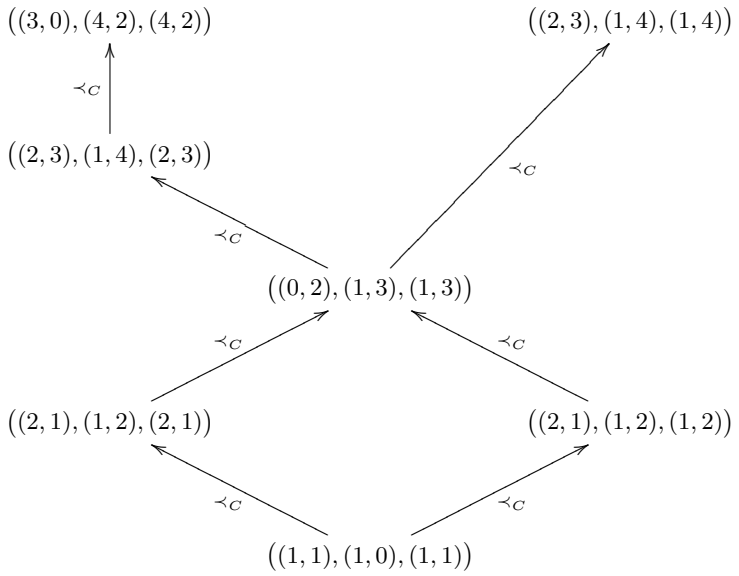


FIGURE 3.1. Hasse diagram of the graph associated with the data in Example 3.1.1

We denote by $\Im_A = \phi(\Upsilon(J_A))$, i.e.,

$$\Im_A = \{(u - v, w) : u - v \in Ker(A), w = setld(u, v)\}.$$

It is clear that the maximal chains, $F_1, \ldots, F_r$, of the image of $\Upsilon(\mathbf{u}, \mathbf{v})$ under ϕ with respect to the order $\prec_C$ over the second components satisfy the following properties:

(1) F_i is totally ordered by the second components with respect to $\prec_C$, for $i = 1, \ldots, r$.
(2) For all $(\alpha, \beta) \in F_i$, $i = 1, \ldots, r$, $A(\beta - \alpha) = A\beta$.

The map ϕ and the above properties allow us to define the notion of test family for $MIP_{A,C}$. This notion is analogous to the concept of test set for a family of single objective integer programs when we have a partial order rather than a total order over $\mathbb{N}^n$ [103]. Test families are instrumental for finding the Pareto-optimal set of each member $MIP_{A,C}(b)$ of the family of multiobjective integer linear programs.

DEFINITION 3.1.1 (Test Family). *A finite collection $\mathcal{G} = \{\mathcal{G}_C^1, \ldots, \mathcal{G}_C^r\}$ of sets in $\mathbb{Z}^n \times \mathbb{Z}_+^n$ is a test family for $MIP_{A,C}$ if and only if:*

(1) *$\mathcal{G}_C^j$ is totally ordered by the second component with respect to $\prec_C$, for $j = 1, \ldots, r$.*

(2) *For all* $(g,h) \in \mathcal{G}_C^j$, $j = 1,\ldots,r$, $A\,(h-g) = A\,h$.
(3) *If* $x \in \mathbb{N}^n$ *is a dominated solution for* $MIP_{A,C}(b)$, *with* $b \in \mathbb{Z}_+^n$, *there is some* $\mathcal{G}_C^j$ *in the collection and* $(g,h) \in \mathcal{G}_C^j$, *such that* $x - g \prec_C x$.
(4) *If* $x \in \mathbb{N}^n$ *is a Pareto-optimal solution for* $MIP_{A,C}(b)$, *with* $b \in \mathbb{Z}_+^n$, *then for all* $(g,h) \in \mathcal{G}_C^j$ *and for all* $j = 1,\ldots,r$ *either* $x - g$ *is infeasible or* $x - g$ *is incomparable to* x.

Given a test family for $MIP_{A,C}$ there is a natural approach for finding the entire Pareto-optimal set. Suppose we wish to solve $MIP_{A,C}(b)$ for which x^* is a feasible solution.

If x^* is dominated then there is some j and $(g,h) \in \mathcal{G}_C^j$ such that $x^* - g$ is feasible and $x^* - g \prec_C x^*$, whereas for the remaining chains there may exist some (g,h) such that $x^* - g$ is feasible but incomparable to x^*. We keep track of all of them.

If x^* is non-dominated, we have to keep it as an element in our current solution set. Then, reducing x^* by the chains in the test family we can only obtain either incomparable feasible solutions, that we maintain in our structure, or infeasible solutions that are discarded.

The above two cases lead us to generate the following set. From x^* we compute the set of incumbent solutions:

$$IS(x^*) := \{y^* : y^* = x^* - g_{j_i}, (g_{j_i}, h_{j_i}) \text{ is the largest element } (g,h) \text{ in the chain } \mathcal{G}_C^i \text{ such that } x^* - g \text{ is feasible }, i = 1,\ldots,r\}.$$

Now, the scheme proceeds recursively on each element of the set $IS(x^*)$. Finiteness of the above scheme is clear since we are generating a search tree with bounded depth (cardinality of the test family) and bounded width, each element in the tree has at most r (number of chains) followers. Correctness of this approach is ensured since any pair of Pareto-optimal solutions must be connected by a reduction chain through elements in the test family (see Theorem 3.1.1 and Corollary 3.1.1).

The above approach assumes that a feasible solution to $MIP_{A,C}(b)$ is known (thus implying that the problem is feasible). Methods to detect infeasibility and to get an initial feasible solution are connected to solving diophantine systems of linear equations, the interested reader is referred to [86] for further details.

The following lemmas help us in describing the geometric structure of a test family for multiobjective integer linear problems.

LEMMA 3.1.1 (Gordan-Dickson Lemma, Theorem 5 in [33]). *If* $P \subseteq \mathbb{N}^n$, $P \neq \emptyset$, *then there exists a minimal subset* $\{p_1,\ldots,p_m\} \subseteq P$ *that is finite*

and unique such that $p \in P$ implies $p_j \leq p$ (component-wise) for at least one $j = 1, \ldots, m$.

LEMMA 3.1.2. *There exists a unique, minimal, finite set of vectors $\alpha_1, \ldots, \alpha_k \in \mathbb{N}^n$ such that the set $\mathcal{L}_C$ of all dominated solutions in all fibers of $MIP_{A,C}$ is a subset of $\mathbb{N}^n$ of the form*

$$\mathcal{L}_C = \bigcup_{j=1}^{k} (\alpha_j + \mathbb{N}^n).$$

PROOF. The set of dominated solutions of all problems $MIP_{A,C}$ is:

$$\mathcal{L}_C = \{\alpha \in \mathbb{N}^n : \exists \beta \in \mathbb{N}^n \text{ with } A\beta = A\alpha \text{ and } \beta \prec_C \alpha\}.$$

Let α be an element in $\mathcal{L}_C$ and β a Pareto-optimal point in the fiber $\pi^{-1}(A\alpha)$ that satisfies $\beta \prec_C \alpha$. Then, for any $\gamma \in \mathbb{N}^n$, $A(\alpha + \gamma) = A(\beta + \gamma)$, $\alpha + \gamma, \beta + \gamma \in \mathbb{N}^n$ and $\beta + \gamma \prec_C \alpha + \gamma$, because the cost matrix, C, has only nonnegative coefficients. Therefore, $\alpha + \gamma$ is a feasible solution dominated by $\beta + \gamma$ in the fiber $\pi^{-1}(A(\alpha + \gamma))$. Then, $\alpha + \gamma \in \mathcal{L}_C$ for all $\gamma \in \mathbb{N}^n$, so, $\alpha + \mathbb{N}^n \subseteq \mathcal{L}_C$. By Lemma 3.1.1 we conclude that there exists a minimal set of elements $\alpha_1, \ldots, \alpha_k \in \mathbb{N}^n$ such that $\mathcal{L}_C = \bigcup_{j=1}^{k} (\alpha_j + \mathbb{N}^n)$. □

Once elements $\alpha_1, \ldots, \alpha_k$ generating $\mathcal{L}_C$ (in the sense of the above result) have been obtained, one can compute the maximal chains of the set $\{\alpha_1, \ldots, \alpha_k\}$ with respect to the partial order $\prec_C$. We denote by $\mathcal{C}_C^1, \ldots, \mathcal{C}_C^\mu$ these maximal chains and set $\mathcal{L}_C^i = \bigcup_{t=1}^{k_i} (\alpha_t^i + \mathbb{N}^n)$, where $\alpha_t^i \in \mathcal{C}_C^i$ for $t = 1, \ldots, k_i$ and $i = 1, \ldots, \mu$. For details about maximal chains, upper bounds on its cardinality and algorithms to compute them for a partially ordered set, the reader is refereed to [5].

It is clear that with this construction we have: $\mathcal{L}_C = \bigcup_{i=1}^{\mu} \mathcal{L}_C^i$.

Next, we describe a finite family of sets $\mathcal{G}_{\prec_C} \subseteq Ker(A) \cap \mathbb{Z}^n$ and prove that it is indeed a test family for $MIP_{A,C}$.

Let $\mathcal{G}_{\prec_C} = \{\mathcal{G}^i_{\prec_C}\}_{i=1}^{\mu}$, being

(12)

$$\mathcal{G}^i_{\prec_C} = \{(g^k_{ij}, h^k_{ij}) = (\alpha^i_j - \beta^k_{ij}, \alpha^i_j), j = 1, \ldots k_i, k = 1, \ldots, m_{ij}\}, i = 1, \ldots, \mu,$$

the maximal chains of $\mathcal{G}_{\prec_C}$ (with respect to the order $\prec_C$ over the second components) and where $\alpha^i_1, \ldots, \alpha^i_{k_i}$ are the unique minimal elements of $\mathcal{L}^i_{\prec_C}$ and $\beta^1_{ij}, \ldots, \beta^{m_{ij}}_{ij}$ the Pareto-optimal solutions to the problem $MIP_{A,C}(A\alpha^i_j)$.

In the next section we give an algorithm that explicitly constructs $\mathcal{G}_{\prec_C}$. Notice that for fixed i, j and k, $g^k_{ij} = (\alpha^i_j - \beta^k_{ij})$ is a point in the subspace $S = \{x \in \mathbb{Q}^n : Ax = 0\}$, i.e., in the 0-fiber of $MIP_{A,C}$. Geometrically we think of $(\alpha^i_j - \beta^k_{ij}, \alpha^i_j)$ as the oriented vector $\overrightarrow{g}^k_{ij} = \overrightarrow{[\beta^k_{ij}, \alpha^i_j]}$ in the $A\alpha^i_j$-fiber of

$MIP_{A,C}$. The vector is directed from the Pareto-optimal point β_{ij}^k to the non-optimal point α_j^i to due to the minimization criterion in $MIP_{A,C}$ which requires us to move away from expensive points. Subtracting the point $\overrightarrow{g}_{ij}^k = \alpha_j^i - \beta_{ij}^k$ to the feasible solution γ gives the new solution $\gamma - \alpha_j^i + \beta_{ij}^k$ which is equivalent to translating $\overrightarrow{g}_{ij}^k$ by a nonnegative integer vector.

Consider an arbitrary fiber of $MIP_{A,C}$ and a feasible lattice point γ in this fiber. For each vector $\overrightarrow{g}_{ij}^k$ in $\mathcal{G}_{\prec_C}$, check whether $\gamma - g_{ij}^k$ is in $\mathbb{N}^n$. At γ draw all such possible translations of vectors from $\mathcal{G}_{\prec_C}$. The head of the translated vector is also incident at a feasible point in the same fiber as γ since g_{ij}^k is in the 0-fiber of $MIP_{A,C}$. We do this construction for all feasible points in all fibers of $MIP_{A,C}$. From Lemma 3.1.2 and the definition of $\mathcal{G}_{\prec_C}$, it follows that no vector $(\alpha_j^i - \beta_{ij}^k, \alpha_j^i)$ in $\mathcal{G}_{\prec_C}$ can be translated by a ν in $\mathbb{N}^n$ such that its tail meets a Pareto-optimal solution on a fiber unless the obtained vector is incomparable to the Pareto-optimal point β_{ij}^k.

THEOREM 3.1.1. *The above construction builds a connected directed graph in every fiber of $MIP_{A,C}$. The nodes of the graph are all the lattice points in the fiber and $(\gamma, \gamma^{'})$ is an edge of the directed graph if $\gamma^{'} = \gamma - g_{ij}^k$ for some i, j and k. Any directed path of this graph is non-increasing with respect to the partial order $\prec_C$.*

PROOF. Pick a fiber of $MIP_{A,C}$ and at each feasible lattice point construct all possible translations of the vector $\overrightarrow{g}_{ij}^k$ from the set $\mathcal{G}_{\prec_C}^i$ as described above. Let α be a lattice point in this fiber. By Lemma 3.1.2, $\alpha = \alpha_j^i + \nu$ for some $i \in \{1, \ldots, t\}$ and $\nu \in \mathbb{Z}_+^n$. Now, since the point $\alpha_k^{'}$ defined as $\alpha_k^{'} = \beta_{ij}^k + \nu$ also lies in the same fiber that α, then $\alpha_k^{'} \prec_C \alpha$ or $\alpha_k^{'}$ and α are incomparable. Therefore, $\overrightarrow{g}_{ij}^k$ translated by $\nu \in \mathbb{N}^n$ is an edge of this graph and we can move along it from α to a point $\alpha^{'}$ in the same fiber, such that $\alpha^{'} \prec_C \alpha$ or α and $\alpha^{'}$ are incomparable. This proves that from every dominated point in the fiber we can reach an improved or incomparable point (with respect to $\prec_C$) in the same fiber by moving along an edge of the graph. □

We call the graph in the b-fiber of $MIP_{A,C}$ built from elements in $\mathcal{G}_{\prec_C}$, the $\prec_C$-skeleton of that fiber.

The reader may note that from each dominated solution α, one can easily build paths to its comparable Pareto-optimal solutions subtracting elements in $\mathcal{G}_{\prec_C}$. Indeed, let β a Pareto-optimal solution in the $A\alpha$-fiber such that β dominates α. Then, let α_i be a minimal element of $\mathcal{L}_C$ such that $\alpha = \alpha_i + \gamma$, with $\gamma \in \mathbb{N}^n$, and let β_i be the Pareto-optimal solution in the $A\alpha_i$-fiber that is comparable to α_i and such that $\beta_i + \gamma$ is comparable to β. Then $\alpha^{'} = \beta_i + \gamma$ is a solution in the $A\alpha$-fiber with $\beta \prec_C \alpha^{'} \prec_C \alpha$. Now, one repeats this process

but starting with $\alpha^{'}$ and β, until $\alpha^{'} = \beta$. Moreover, the case where α and β are incomparable reduces to the previous one by finding a path from α to any intermediate point $\beta^{'}$ that compares with β. This analysis leads us to the following result.

COROLLARY 3.1.1. *In the $\prec_C$-skeleton of a fiber there exists a directed path from every feasible point α to each Pareto-optimal point, β, in the same fiber. The vectors of objective function values of successive points in the path do not increase componentwise from α to β.*

COROLLARY 3.1.2. *The family $\mathcal{G}_{\prec_C}$ is the unique minimal test family for $MIP_{A,C}$. It depends only on the matrix A and the cost matrix C.*

PROOF. By definition of $\mathcal{G}_{\prec_C}$, the conditions 1. and 2. of Definition 3.1.1 are satisfied. From Theorem 3.1.1 it follows that conditions 3. and 4. are also satisfied, so $\mathcal{G}_{\prec_C}$ is a test family for $MIP_{A,C}$. Minimality is due to the fact that removing any element (g_{ij}^k, h_{ij}^k) from $\mathcal{G}_{\prec_C}$ results in $\mathcal{G}_{\prec_C} \backslash \{(g_{ij}^k, h_{ij}^k)\}$. However, this new set is not a test family since no oriented vector in $\mathcal{G}_{\prec_C} \setminus \{(g_{ij}^k, h_{ij}^k)\}$ can be translated through a nonnegative vector in $\mathbb{N}^n$ such that its tail meets α_j^i. It is clear by definition that $\mathcal{G}_{\prec_C}$ depends only on A and C. □

EXAMPLE 3.1.2. *Let $MIP_{A,C}$ be the family of multiobjective problems, with the following constraints and objective function matrices:*

$$A = \begin{bmatrix} 2 & 2 & -1 & 0 \\ 0 & 2 & 0 & 1 \end{bmatrix}, \qquad C = \begin{bmatrix} 10 & 1 & 0 & 0 \\ 1 & 10 & 0 & 0 \end{bmatrix}.$$

Let (x_1, x_2, s_1, s_2) be the vector of variables, where s_1 and s_2 are slack variables. In this example, using the order $\prec_C^s$ (see Remark 24), $\mathcal{G}_{\prec_C} = \{\mathcal{G}^1_{\prec_C}, \mathcal{G}^2_{\prec_C}\}$, where $\mathcal{G}^1_{\prec_C} = \{\overrightarrow{g}^1_1 = ((0,1,2,-1),(0,1,2,0))$,
$\overrightarrow{g}^1_2 = ((-1,1,0,-2),(0,1,0,0))\}$ and $\mathcal{G}^2_{\prec_C} = \{\overrightarrow{g}^2_1 = ((1,0,2,0),\ (1,0,2,0))$,
$\overrightarrow{g}^2_2 = ((1,-1,0,2),(1,0,0,2))\}$.

Figure 3.2 shows, on the (x_1, x_2)-plane, the $\prec_C$-skeleton of the fiber corresponding to the right-hand side vector $(17, 11)^t$. In the box over the graph of the $\prec_C$-skeleton, we show the second components of the elements of $\mathcal{G}_{\prec_C}$. The reader may note that in the graph, the arrows have opposite directions due to the fact that the directed paths (improving solutions) are built subtracting the elements in $\mathcal{G}_{\prec_C}$. We describe how to compute the sets $\mathcal{G}^1_{\prec_C}$ and $\mathcal{G}^2_{\prec_C}$ in Section 3.2.

Given $\mathcal{G}_{\prec_C}$, there are several ways to build a path from each feasible point in a fixed fiber to any Pareto-optimal solution. However, there is a canonical way to do it: Fix σ a permutation of the set $\{1, \ldots, \mu\}$ and subtract from the

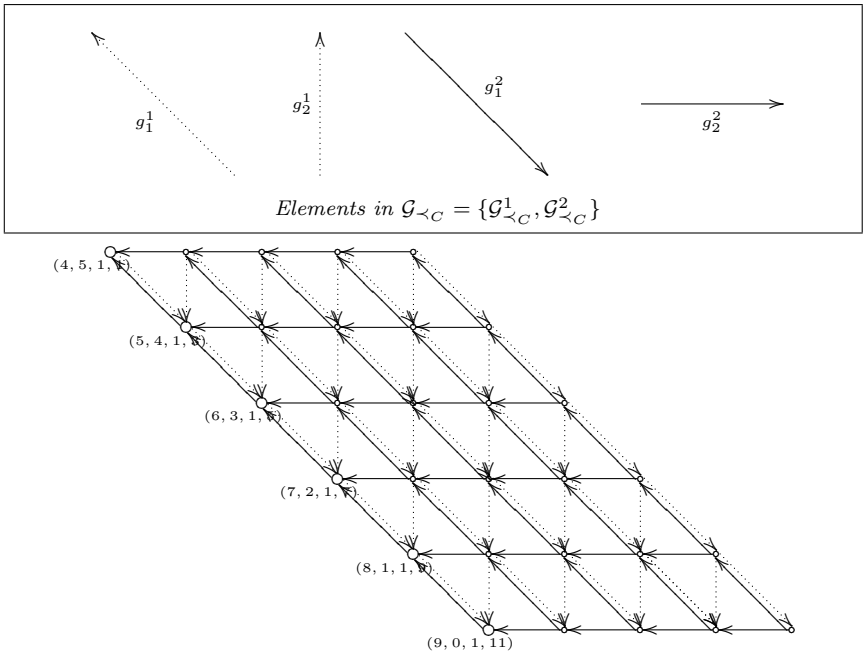


FIGURE 3.2. The $\prec_C$-skeleton of the $(17, 11)^t$-fiber of $MIP_{A,C}$ projected on the x_1, x_2-plane.

initial point the elements of $\mathcal{G}^{\sigma(i)}_{\prec_C}$, for $i = 1, \ldots, \mu$. Add this element to an empty list. After each substraction by elements in $\mathcal{G}^{\sigma(i)}_{\prec_C}$, $i = 1, \ldots, \mu$, remove from the list those elements dominated by the new element. We prove in Section 3 that this result does not depend on the permutation σ.

EXAMPLE 3.1.2 (Continuation). *This example shows the above mentioned different ways to compute paths from dominated solutions to any Pareto-optimal solution. The vector* $(9, 4, 9, 3)$ *is a feasible solution for* $MIP_{A,C}$ *in the* $(17, 11)^t$*-fiber. Figure 3.3 shows the sequence of Pareto-optimal points obtained from the feasible point* $(9, 4, 9, 3)$ *using the permutation* $\sigma_1 = (1, 2)$ *(on the left) and using* $\sigma_2 = (2, 1)$ *(on the right).*

REMARK 3.1.1. *Let* $\prec_C$ *be the partial order induced by* C*. Then, a directed path from a dominated point* α *to each Pareto-optimal point* β *in a fiber, applying the above method, cannot pass through any lattice point in this fiber more than* μ *times (recall that* μ *is the number of maximal chains in* $\mathcal{G}_{\prec_C}$*). This implies that obtaining the Pareto-optimal solutions of a given* $MIP_{A,C}$ *using* $\mathcal{G}_{\prec_C}$ *cannot cycle.*

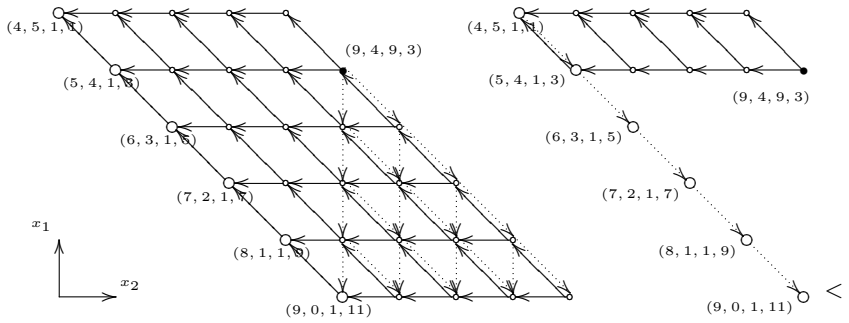


FIGURE 3.3. Different ways to compute paths from $(9, 4, 9, 3)$ to the Pareto-optimal solutions in its fiber.

3.2. Test families and Partial Gröbner bases

In the previous section we motivated the importance of having a test family for $MIP_{A,C}$ since this structure allows us to obtain the entire set of Pareto-optimal solutions of the above family of multiobjective integer programs (when the right-hand side varies). Our goal in this section is to provide the necessary tools to construct test families for any multiobjective integer problem. Our construction builds upon an extension of Gröbner bases on partial orders.

In order to introduce this structure we define the reduction of a pair $(g, h) \in \mathbb{Z}^n \times \mathbb{Z}^n_+$ by a finite set of ordered pairs in $\mathbb{Z}^n \times \mathbb{Z}^n_+$. Given is a collection $\mathcal{G}_C \subseteq \mathbb{Z}^n \times \mathbb{Z}^n_+$ where $\mathcal{G}_C = \{(g_1, h_1), \ldots, (g_l, h_l) : h_{k+1} \prec_C h_k, k = 1, \ldots, l-1\}$.

The reduction of (g, h) by $\mathcal{G}_C$ consists of the process described in Algorithm 9. The above reduction process extends to the case of a finite collection of ordered sets of pairs in $\mathbb{Z}^n \times \mathbb{Z}^n_+$ by establishing the sequence in which the sets of pairs are considered. We denote by $pRem((g, h), \mathcal{G})_\sigma$ the reduction of the pair (g, h) by the family $\mathcal{G} = \{\mathcal{G}_i\}_{i=1}^t$ for a fixed sequence of indices σ.

From now on, we denote by $pRem((g, h), \mathcal{G})$ the set of remainders of (g, h) by the family $\mathcal{G} = \{\mathcal{G}_i\}_{i=1}^t$ for the natural sequence of indices $(1, \cdots, t)$, i.e. when σ is the identity.

The reduction of a pair that represents a feasible solution, by a test family, gives the entire set of Pareto-optimal solutions. In order to obtain that test family, we introduce the notion of geometric p-Gröbner basis. This name has been motivated by the fact that when the ordering in $\mathbb{N}^n$ is induced by a single cost vector, a Gröbner basis is a test set for the family of integer programs $IP_{A,c}$ (see [30] or [103] for extended details). In the single objective case the Buchberger algorithm computes a Gröbner basis. However, in the multiobjective case the cost matrix induces a partial order, so division or the

Algorithm 9: Partial reduction algorithm

input : $R = \{(g,h)\}$, $S = \{(g,h)\}$,
$\mathcal{G}_C = \{(g_1,h_1),\ldots,(g_l,h_l) : h_{k+1} \prec_C h_k, k = 1,\ldots,l-1\}$
Set $i := 1$, $S_o = \{\}$.
repeat
  for $(\tilde{g},\tilde{h}) \in S \setminus S_o$ **do**
    while $\tilde{h} - h_i \geq 0$ **do**
      if $\tilde{h} - g_i$ *and* $\tilde{h} - \tilde{g}$ *are comparable by* $\prec_C$ **then**
        $R_o = \{(\tilde{g} - g_i, \max_{\prec_C}\{\tilde{h} - g_i, \tilde{h} - \tilde{g}\})\}$
      else
        $R_o = \{(\tilde{g} - g_i, \tilde{h} - g_i), (\tilde{g} - g_i, \tilde{h} - \tilde{g})\}$
      end
      For each $r \in R_o$ and $s \in R$:
      if $r \prec_C s$ **then**
        $R = R \backslash \{s\}$;
      end
      $S = R_o$.
      $R = R \cup R_o$.
      $S_o = S_o \cup \{(\tilde{g},\tilde{h})\}$.
    end
  end
  $i = i + 1$;
until $i \leq t$;
output: R, the partial reduction set of (g,h) by $\mathcal{G}_C$

Buchberger algorithm are not applicable. Using the above reduction algorithm (Algorithm 9) we present an "à la Buchberger" algorithm to compute the so called geometric p-Gröbner basis to solve MOILP problems.

Definition 3.2.1 (Partial Gröbner basis). *A family $\mathcal{G} = \{\mathcal{G}_1,\ldots,\mathcal{G}_t\} \subseteq \Im_A$ is a geometric partial Gröbner basis (geometric p-Gröbner basis) for the family of problems $MIP_{A,C}$, if $\mathcal{G}_1,\ldots,\mathcal{G}_t$ are the maximal chains for the partially ordered set $\bigcup_{i=1}^{t} \mathcal{G}_i$ and for any $(g,h) \in \mathbb{Z}^n \times \mathbb{Z}^n_+$ with $h - g \geq 0$:*

$$g \in Ker(A) \Longleftrightarrow pRem((g,h),\mathcal{G})_\sigma = \{0\}.$$

for any sequence σ.

A geometric p-Gröbner basis is said to be reduced if every element in each maximal chain cannot be obtained by reducing any other element of the same chain.

Given a geometric p-Gröbner basis, computing a reduced geometric p-Gröbner basis is done by deleting the elements that can be reduced by other elements in the basis. After the removing process, the family is a geometric p-Gröbner basis having only non redundant elements. It is easy to see that

the reduced geometric p-Gröbner basis for $MIP_{A,C}$ is unique and minimal, in the sense that no element can be removed from it maintaining the geometric p-Gröbner basis structure.

This definition coincides with the notion of p-Gröbner bases (Definition 2.2.1) for the ideal $\Im_A$ induced by A, once we fix the partial order, $\prec_C$, induced by C.

In the following, we present algorithms to solve multiobjective problems analogous to the methods that solve the single objective case, using usual Gröbner basis. These methods are based on computing the reduction of a feasible solution by the basis. The key for that result is the fact that the reduction of any pair of feasible solutions is the same, therefore the algorithm is valid for any initial feasible solution. After the following theorem, Lemma 3.2.1 ensures the same statement for the multiobjective case and geometric p-Gröbner bases.

THEOREM 3.2.1. *Let $\mathcal{G}$ be the reduced geometric p-Gröbner basis for $MIP_{A,C}$ and α a feasible solution for $MIP_{A,C}(A\alpha)$. Then,*

$$pRem((\alpha,\alpha),\mathcal{G})_\sigma = pRem((\alpha,\alpha),\mathcal{G})_{\sigma'},$$

for any sequences σ and σ'.

PROOF. We first observe that the elements in $pRem((\alpha,\alpha),\mathcal{G})_\sigma$ are of the form (β,β). Indeed, since the first step of Algorithm 9 reduces the element (α,α) then $\tilde{h}-\tilde{g}=\alpha-\alpha=0$. Therefore, $\tilde{h}-\tilde{g}$ is always dominated by $\tilde{h}-g_i$ because $0 \prec_C \tilde{h}-g_i$, so that the remainders are of the form $(\alpha-g_i,\alpha-g_i)$.

On other hand, let (β,β) be an element in $pRem((\alpha,\alpha),\mathcal{G})_\sigma$, then $\alpha-\beta \in Ker(A)$ and by Definition 3.2.1, $pRem((\alpha-\beta,\alpha),\mathcal{G})_{\sigma'} = pRem((\alpha-\beta,\beta),\mathcal{G})_{\sigma'} = \{0\}$ for any σ'. □

The above result ensures that without loss of generality reductions of elements of the form (α,α) by p-Gröbner bases are independent of the permutation of indices used. Therefore, we do not make reference to σ in the notation referring always to the natural sequence $\sigma=(1,\ldots,t)$.

LEMMA 3.2.1. *Let $\mathcal{G}$ be the reduced geometric p-Gröbner basis for $MIP_{A,C}$ and α_1,α_2 two different feasible solutions in the same fiber of $MIP_{A,C}$. Then, $pRem((\alpha_1,\alpha_1),\mathcal{G}) = pRem((\alpha_2,\alpha_2),\mathcal{G})$.*

PROOF. Let $(\beta,\beta) \in pRem((\alpha_1,\alpha_1),\mathcal{G})$, then since $A\alpha_1 = A\alpha_2$, β is in the same fiber that α_2. Next, since β cannot be reduced, then $(\beta,\beta) \in pRem((\alpha_2,\alpha_2),\mathcal{G})$. □

The following theorem states the relationship between the three structures introduced before: test families, reduced p-Gröbner bases and the family $\mathcal{G}_{\prec_C}$.

THEOREM 3.2.2. *The reduced p-Gröbner basis for $MIP_{A,C}$ is the unique minimal test family for $MIP_{A,C}$. Moreover, $\mathcal{G}_{\prec_C}$, introduced in (12), is the reduced p-Gröbner basis for $MIP_{A,C}$.*

PROOF. Let $\mathcal{G} = \{\mathcal{G}_1, \ldots, \mathcal{G}_t\}$ be the reduced p-Gröbner basis for $MIP_{A,C}$. We have to prove that $\mathcal{G}$ satisfies the four conditions in Definition 3.1.1. By definition of geometric p-Gröbner basis, it is clear that each $\mathcal{G}_i$ is totally ordered by its second component with respect to $\prec_C$ (Condition 1). Condition 2 follows because for each i and for each $(g,h) \in \mathcal{G}_i \subseteq \mathbb{Z}^n \times \mathbb{Z}^n_+$, clearly $pRem((g,h),\mathcal{G}) = \{0\}$, so $g \in Ker(A)$ and then $A(h-g) = Ah$.

Now, let $x \in \mathbb{N}^n$ be a dominated solution for $MIP_{A,C}(b)$. Then, there is a Pareto-optimal solution, β, such that $\beta \prec_C x$. By Lemma 3.2.1, $pRem((x,x),\mathcal{G}) = pRem((\beta,\beta),\mathcal{G})$, and by construction of the set of partial remainders, $\beta \in pRem((\beta,\ \beta),\ \mathcal{G})$, thus $x \notin pRem((x,x),\mathcal{G})$. This implies that there exists $(g,h) \in \mathcal{G}_i$, for some $i = 1, \ldots, t$, such that $x - g \prec_C x$. This proves condition 3.

On the other hand, if x is a Pareto-optimal solution for $MIP_{A,C}(b)$, $x \in pRem((x,\ x),\ \mathcal{G})$, then there exists no (g,h) in any $\mathcal{G}_i$ such that $x - g \prec_C x$. Therefore, for every i and for each $(g,h) \in \mathcal{G}_i$, either $x - g$ is infeasible or incomparable to x, which proves condition 4.

Minimality is due to the fact that removing an element from the reduced geometric p-Gröbner basis, that is the minimal geometric partial Gröbner basis that can be built for $MIP_{A,C}$ we cannot guarantee to have a test family because it may exist a pair $(g,h) \in \mathbb{Z}^n \times \mathbb{Z}^n_+$ with $g \in Ker(A)$ that cannot be reduced to the zero set.

Finally, the second statement of the theorem follows from Corollary 3.1.2. □

Next, we describe an extended algorithm to compute a geometric p-Gröbner basis for $\Im_A$, with respect to the partial order induced by C. This algorithm is the geometrical transcription of Buchberber algorithm for p-Gröbner bases (Algorithm (5)). First, we need to define the geometrical equivalent to the S-polynomials (6). For any $(g,h),(g',h')$ in $\mathbb{Z}^n \times \mathbb{Z}^n_+$ we denote by $S^1((g,h),(g',h'))$ and $S^2((g,h),(g',h'))$ the pairs

$$S^1((g,h),(g',h')) = \begin{cases} (g-g'-2(h-h'),\gamma+g-2h) & \text{if } \gamma+g-2h \prec_C \gamma+g'-2h' \\ (g'-g-2(h'-h),\gamma+g'-2h') & \text{if } \gamma+g'-2h' \prec_C \gamma+g-2h \\ (g-g'-2(h-h'),\gamma+g-2h) & \text{if } \gamma+g'-2h' \text{ and } \gamma+g-2h \text{ are incomparable} \end{cases}$$

and

$$S^2((g,h),(g^{'},h^{'})) = \begin{cases} (g-g^{'}-2(h-h^{'}),\gamma+g-2h) & \text{if } \gamma+g-2h \prec_C \gamma+g^{'}-2h^{'} \\ (g^{'}-g-2(h^{'}-h),\gamma+g^{'}-2h^{'}) & \text{if } \gamma+g^{'}-2h^{'} \prec_C \gamma+g-2h \\ (g^{'}-g-2(h^{'}-h),\gamma+g^{'}-2h^{'}) & \text{if } \gamma+g^{'}-2h^{'} \text{ and } \gamma+g-2h \\ & \text{are incomparable} \end{cases}$$

where $\gamma \in \mathbb{N}^n$ and $\gamma_i = \max\{h_i, h_i^{'}\}$, $i = 1,\ldots,n$.

The pairs $S^1((g,h),(g^{'},h^{'}))$ and $S^2((g,h),(g^{'},h^{'}))$ are called $1-Svector$ and $2-Svector$ of (g,h) and $(g^{'},h^{'})$, respectively. The reader may note that $S^1((g,h),(g^{'},h^{'}))$ and $S^2((g,h),(g^{'},h^{'}))$ coincide provided that the resulting pairs are comparable under $\prec_C$, whereas they correspond with the two possible choices of the new pair in the case when the vectors $\gamma+g^{'}-2h^{'}$ and $\gamma+g-2h$ are incomparable.

The following result is the geometric transcription of Theorem 2.2.1. Actually, the proof for this theorem would begin identifying vectors with binomials.

THEOREM 3.2.3 (Extended geometric Buchberger's criterion). *Let $\mathcal{G} = \{\mathcal{G}_1,\ldots,\mathcal{G}_t\}$ with $\mathcal{G}_i \subseteq \Im_A$ for all $i = 1,\ldots,t$, be the maximal chains of the partially ordered set $\{g_i : g_i \in \mathcal{G}_i,$ for some $i = 1,\ldots,t\}$, and such that $\mathcal{G}^*$, the polynomial transcription of $\mathcal{G}$, is a system of generators of I_A. Then the following statements are equivalent:*

(1) *$\mathcal{G}$ is a p-Gröbner basis for the family $MIP_{A,C}$.*
(2) *For each $i,j = 1,\ldots,t$ and $(g,h) \in \mathcal{G}_i$, $(g^{'},h^{'}) \in \mathcal{G}_j$, $pRem(S^k((g,h),(g^{'},h^{'})),\mathcal{G}) = \{0\}$, for $k =, 1, 2$.*

PROOF. The proof follows from the analogy between the algebraic and the geometric notion of p-Gröbner basis and Theorem 2.2.1. □

This criterion (the one in Theorem 3.2.3) allows us to describe a geometric algorithm which constructs a geometric p-Gröbner basis $\mathcal{G}_C$ for $MIP_{A,C}$, and therefore a test family for that family of multiobjective problems.

The first approach to compute a p-Gröbner basis for a family of multiobjective programs is based on Conti and Traverso method for the single objective case [30]. In Chapter 3 was described this algorithm using an algebraic language. Here, we recall the same algorithm using the geometric notation.

Given the program $MIP_{A,C}(b)$, we consider the associated extended multiobjective program, $EMIP_{A,C}(b)$ as the problem $MIP_{\widetilde{A},\widetilde{C}}(b)$ where

$$\widetilde{A} = \left(\begin{array}{c|c|c} & -1 & \\ Id_m & \vdots & A \\ & -1 & \end{array}\right) \in \mathbb{Z}^{m\times(m+1+n)},$$

Algorithm 10: Partial Buchberger algorithm I

input : $F_1 = \{M_0, M_1, \ldots, M_n\}$ and $F_2 = \{P_0, P_1, \ldots, P_n\}$,
$M_i = (a_{1i} - \min\{0, \min_j\{a_{ji}\}\}, \ldots, a_{mi} - \min\{0, \min_j\{a_{ji}\}\}, -\min\{0, \min_j\{a_{ji}\}\}, 0, \overset{n}{\ldots}, 0)$ $(i > 0)$
$P_i = (0, \overset{m+1}{\ldots}, 0|e_i) \in \mathbb{N}^{m+n+1}$ $(i > 0)$
$M_0 = (1, \overset{m+1}{\ldots}, 1, 0, \overset{n}{\ldots}, 0)$
$P_0 = (0, \overset{n+m+1}{\ldots}, 0)$.

repeat
  Compute, $\mathcal{G}_1, \ldots, \mathcal{G}_t$, the maximal chains for $\mathcal{G} = \phi(\Upsilon(F_1, F_2))$.
  for $i, j \in \{1, \ldots, t\}$, $i \neq j$, *and each pair* $(g, h) \in \mathcal{G}_i$, $(g', h') \in \mathcal{G}_j$ **do**
    Compute $R^k = pRem(S^k((g, h), (g^{'}, h^{'})), \mathcal{G})$, $k = 1, 2$.
    if $R^k = \{0\}$ **then**
      Continue with other pair.
    else
      Add $\phi(\Upsilon(r))$ to $\mathcal{G}$, for each $r \in R^k$.
    end
  end
until $R^k = \{0\}$ *for every pairs* ;
output: $\mathcal{G} = \{\mathcal{G}_1, \ldots, \mathcal{G}_Q\}$ geometric p-Gröbner basis for $\Im_A$ with respect to $\prec_C$.

$\widetilde{C} = (M \cdot \mathbf{1}|C) \in \mathbb{Z}^{(m+1+n)\times k}$, Id_m stands for the $m \times m$ identity matrix, M is a large constant and $\mathbf{1}$ is the $(m+1) \times k$ matrix whose components are all 1. This problem adds $m+1$ new variables, whose weights in the multiobjective function are big, and so, solving this extended minimization program allows us to solve directly the initial program $MIP_{A,C}$. Indeed, any feasible solution to the original problem is a feasible solution to the extended problem with the first m components equal to zero, so any feasible solution of the form $(0, \overset{m+1}{\ldots}, 0, \alpha_1, \ldots, \alpha_n)$ is non-dominated, upon the order $\prec_{\widetilde{C}}$, by any solution without zeros in the first m components. Then, computing a geometric p-Gröbner basis for the extended program using the partial Buchberger Algorithm (Algorithm 10), allows us detecting infeasibility of the original problem. Furthermore, a trivial feasible solution, $\widetilde{\mathbf{x}}_0 = (b_1, \ldots, b_m, 0, \overset{n+1}{\ldots}, 0)$, is known and the initial set of generators for $\Im_A$ is given by $\{\{M_i - P_i, M_i\} : i = 0 \ldots, n\}$ where $M_i = (a_{1i} - \min\{0, \min_j\{a_{ji}\}\}, \ldots, a_{mi} - \min\{0, \min_j\{a_{ji}\}\}, -\min\{0, \min_j\{a_{ji}\}\}, 0, \overset{n}{\ldots}, 0)$, $P_i = (0, \overset{m+1}{\ldots}, 0|e_i)$, for all $i = 1, \ldots, n$, $M_0 = (1, \overset{m+1}{\ldots}, 1, 0, \overset{n}{\ldots}, 0)$ and $P_0 = \mathbf{0}$, $M_i, P_i, M_0, P_0 \in \mathbb{Z}_+^{n+m+1}$ (see [2] for further details). Then, we can state the following result.

THEOREM 3.2.4. *Let* $\mathcal{G} = \{\mathcal{G}_i\}_{i=1}^t$ *be a p-Gröbner basis for* $EMIP_{A,C}$ *and* $\mathbf{b} = (b_1, \ldots, b_m)$. *The entire set of Pareto-optimal solutions for* $MIP_{A,C}(\mathbf{b})$ *consists of of the vectors* $\alpha = (\alpha_1, \ldots, \alpha_n)$ *such that* $(0, \overset{m+1}{\ldots}, 0, \alpha_1, \ldots, \alpha_n) \in$

$pRem(((\mathbf{b}, 0, \overset{n+1}{\ldots}, 0), (\mathbf{b}, 0, \overset{n+1}{\ldots}, 0)), \mathcal{G})$. *Moreover, if there is no* α' *in the set* $pRem(((\mathbf{b}, 0, \overset{n+1}{\ldots}, 0), (\mathbf{b}, 0, \overset{n+1}{\ldots}, 0)), \mathcal{G})$ *whose* $m+1$ *first components are zero* $MIP_{A,C}(b)$ *is infeasible.*

PROOF. Let α be a vector obtained by successive reductions over $\mathcal{G}$. It is clear that α is feasible because $((\mathbf{0}, \alpha), (\mathbf{0}, \alpha))$ is in the set of remainders of $((\mathbf{b}, \mathbf{0}), (\mathbf{b}, \mathbf{0}))$ by $\mathcal{G}$ and then, in the same fiber. Besides, α is a Pareto-optimal solution because $\mathcal{G}$ is a test family for the problem (Theorem 3.2.2).

Now, if β^* is a Pareto-optimal solution, by Lemma 3.2.1, $pRem(\ (\ (\mathbf{0}, \beta^*), (\mathbf{0}, \beta^*)\ ), \mathcal{G})) = pRem(((\mathbf{0}, \mathbf{b}), (\mathbf{0}, \mathbf{b})), \mathcal{G}))$, but since β^* is a Pareto-optimal solution, it cannot be reduced so $((\mathbf{0}, \beta^*), (\mathbf{0}, \beta^*)) \in pRem(\ (\ (\mathbf{0}, \beta^*), (\mathbf{0}, \beta^*)\ ), \mathcal{G}))$, and then, $((\mathbf{0}, \beta^*), (\mathbf{0}, \beta^*))$ also belongs to the list of partial remainders of $((\mathbf{b}, \mathbf{0}), (\mathbf{b}, \mathbf{0}))$ by $\mathcal{G}$. □

Hoşten and Sturmfels [60] improved the method by Conti and Traverso to solve single-objective programs using standard Gröbner bases. The method described in Chapter 3 (Algorithm (8)) and the geometric approach differ just in the two last steps (Step 3 and Step 4), since the first step (compute an initial feasible solution) is done in term of vectors in both approaches and the second steps must be done in terms of ideals in both approaches. Moreover, the last two steps consist of computing a p-Gröbner basis (in this case a geometric p-Gröbner basis) and of partially reducing the initial feasible solution by that basis. The first of this procedures has been already described in a geometric language (Algorithm (9)). Computing a geometric Gröbner basis can be done as in the algebraic case, using the Extended Buchberger criterion (Theorem 3.2.3). A pseudocode for the extended Buchberger algorithm for computing geometric p-Gröbner bases is described in Algorithm (11). Then the algorithm is the following: (1) computing an initial basis for the polynomial toric ideal $I_A = \langle x^u - x^v : u - v \in Ker(A)\rangle$, that we can identify with J_A (Algorithm 7, Chapter 2). After that procedure, and identifying I_A with $\Im_A$, we have a initial system of generators for $\Im_A$. Then, with this set of generators, compute a geometric partial Gröbner basis using Algorithm (11).

Algorithm (12) summarizes the complete procedure to solve $MIP_{A,C}$ (b) using geometric p-Gröbner bases.

There are some interesting cases where our methodology is highly simplified due to the structure of the set of constraints. One of these cases is when the dimension of the set of constraints is $n-1$. The next remark explains how the algorithm simplifies in this case.

REMARK 3.2.1. *Let* A *be a* $m \times n$ *integer matrix with rank* $n-1$. *Then, since* $dim(Ker(A)) = 1$, *the system of generators for* $\Im_A$ (**Step 2**) *has just one*

Algorithm 11: `pgrobner`(F_1, F_2)

input : $F_1 = \{M_1, \ldots, M_s\}$ and $F_2 = \{P_1, \ldots, P_s\}$.
repeat
  Compute, $\mathcal{G}_1, \ldots, \mathcal{G}_t$, the maximal chains for $\mathcal{G} = \phi(\Upsilon(F_1, F_2))$.
  for $i, j \in \{1, \ldots, t\}$, $i \neq j$, *and each pair* $(g, h) \in \mathcal{G}_i$, $(g', h') \in \mathcal{G}_j$
  do
    Compute $R^k = pRem(S^k((g, h), (g^{'}, h^{'})), \mathcal{G})$, $k = 1, 2$.
    if $R^k = \{0\}$ **then**
      Continue with other pair.
    else
      Add $\phi(\Upsilon(r))$ to $\mathcal{G}$, for each $r \in R^k$.
    end
  end
until $R^k = \{0\}$ *for every pairs* ;
output: $\mathcal{G} = \{\mathcal{G}_1, \ldots, \mathcal{G}_Q\}$ p-Gröbner basis for $MIP_{A,C}$.

Algorithm 12: Pareto-optimal solutions computation for $MIP_{A,C}(b)$

input : $MIP_{A,C}(b)$
  Step 1. : Compute an initial feasible solution, α_o, for $MIP_{A,C}(b)$.
  Step 2. : Compute a system of generators for $\Im_A$:
    $\{\{u_i, v_i\} : i = 1, \ldots, s\}$, using `setofgenerators(`$A$`)`.
  Step 3. : Compute the partial reduced Gröbner basis for
      $MIP_{A,C}$,
      $\mathcal{G}_C = \{\mathcal{G}_1, \ldots, \mathcal{G}_t\}$, using `pgrobner(` F_1, F_2 `)`, where
    $F_1 = \{u_i : i = 1, \ldots, r\}$
      and $F_2 = \{v_i : i = 1, \ldots, r\}$.
  Step 4. : Calculate the set of partial remainders:
    $R := pRem(\alpha_o, \mathcal{G}_C)$.
output: Pareto-optimal Solutions : R.

element, (g, h)*, and the p-Gröbner basis* (**Step 3**) *is the family* $\mathcal{G} = \{\{(g, h)\}\}$ *because no Svector appears during the computation of the Buchberger algorithm. In this case, Pareto-optimal solutions are obtained as partial remainders of an initial feasible solution* (α, α) *by* (g, h)*, i.e., the entire set of Pareto-optimal solutions is a subset of* $\Gamma = \{\alpha - \lambda g : \lambda \in \mathbb{Z}_+\}$*. More explicitly, the set of Pareto-optimal solutions for* $MIP_{A,C}(b)$ *is the set of minimal elements (with respect to* $\prec_C$*) of* Γ*.*

To illustrate the above approach, we solve Example 2.3.1 with the geometric approach.

EXAMPLE 3.2.1.

$$\begin{array}{ll} \min & \{10x+y, x+10y\} \\ s.t. & \\ & \begin{array}{rl} 2x+2y & \geqslant 17 \\ 2y & \leqslant 11 \\ x & \leqslant 10 \\ x,y & \in \mathbb{Z}_+ \end{array} \end{array} \tag{13}$$

Transforming the problem to the standard form results in:

$$\begin{array}{ll} \min & \{10x+y+0z+0t+0q, x+10y+0z+0t+0q\} \\ s.t. & \\ & \begin{array}{rl} 2x+2y-z & = 17 \\ 2y+t & = 11 \\ x+q & = 10 \\ x,y,z,t,q & \in \mathbb{Z}_+ \end{array} \end{array} \tag{14}$$

Step 1. : *Feasible solution for* $MIP_{A,C}(b)$*:* $u = (9,4,9,3,1)$.

Step 2. : *Following the steps of Algorithm 7:*

(1) *Basis for* $Ker(A)$ *:* $\mathcal{B} := \{(0,1,2,-2,0),(-1,0,-2,0,1)\}$.

(2) *LLL-reduced basis for* $\mathcal{B}$*:*

$$\mathcal{B}_{red} := \mathcal{B} := \{(-1,0,-2,0,1),(-1,1,0,-2,1)\}.$$

(3) $J_0 := \langle x^{u+} - x^{u-} : u \in \mathcal{B}_{red}\rangle = \langle x_5 - x_1x_3^2, x_2x_5 - x_1x_4^2\rangle$

(4) $J_{i+1} := (J_i : x_i^\infty)$

(a) $\widetilde{\mathcal{G}}_0 := \{x_5 - x_1x_3^2, x_2x_5 - x_1x_4^2, x_2x_3^2 - x_4^2\} \Rightarrow J_1 := \langle x_5 - x_1x_3^2, x_2x_5 - x_1x_4^2, x_2x_3^2 - x_4^2\rangle$

(b) $\widetilde{\mathcal{G}}_1 := \{x_5 - x_1x_3^2, x_2x_5 - x_1x_4^2, x_2x_3^2 - x_4^2\} \Rightarrow J_2 := \langle x_5 - x_1x_3^2, x_2x_5 - x_1x_4^2, x_2x_3^2 - x_4^2\rangle$

(c) $\widetilde{\mathcal{G}}_2 := \{x_5 - x_1x_3^2, x_2x_5 - x_1x_4^2, x_2x_3^2 - x_4^2\} \Rightarrow J_3 := \langle x_5 - x_1x_3^2, x_2x_5 - x_1x_4^2, x_2x_3^2 - x_4^2\rangle$

(d) $\widetilde{\mathcal{G}}_3 := \{x_5 - x_1x_3^2, x_2x_5 - x_1x_4^2, x_2x_3^2 - x_4^2\} \Rightarrow J_4 := \langle x_5 - x_1x_3^2, x_2x_5 - x_1x_4^2, x_2x_3^2 - x_4^2\rangle$

(5) $I_A = \langle x_5 - x_1x_3^2, x_2x_5 - x_1x_4^2, x_2x_3^2 - x_4^2, x_1x_3^2 - 1\rangle \mapsto$
$\Im_A = \langle\{\big((1,0,0,0,1),(0,1,0,2,0)\big), \big((1,0,2,0,0),(0,0,0,0,1)\big), \big((0,1,2,0,0),(0,0,0,2,0)\big)\}$

Step 3. : *Computing a p-Gröbner basis* $MIP_{A,C}$*, using the order* $\prec_C^s$ *(Remark 24), and following Algorithm 11 we obtain* $\mathcal{G}$*, whose maximal chains are:*

$\mathcal{G}_1 = \{\big((0,1,2,0,0),\ (0,0,0,2,0),\ (0,1,2,0,0)\big),\ \big((0,1,0,0,2),\ (2,0,2,2,0),\ (0,1,0,0,2)\big),\ \big((0,1,0,0,1),\ (1,0,0,2,0),\ (0,1,0,0,1)\big)\}$

and

$\mathcal{G}_2 = \{\big((1,0,0,4,0),\ (0,2,2,0,1),\ (1,0,0,4,0)\big),\ \big((1,0,2,0,0),\ (0,0,0,0,1),\ (1,0,2,0,0)\big),\ \big((1,0,0,2,0),\ (0,1,0,0,1),\ (1,0,0,2,0)\big)\}$.

Step 4. : *Partial remainders: Reducing first by* $\mathcal{G}_1$*:*

$pRem((9,4,9,3,1),\mathcal{G}_1) = \{(9,0,1,11,1)\}$. *All the steps to obtain this remainder is shown in the Figure 3.4 (projection in the variables* x *and* y*). The blue dots are those final remainders since they are not dominated by any of the others, the grey dots are those that have been discarded by the domination criterion and the one with a red circumference is the initial point for this reduction. The blue dots here will be the initial points for the next step.*

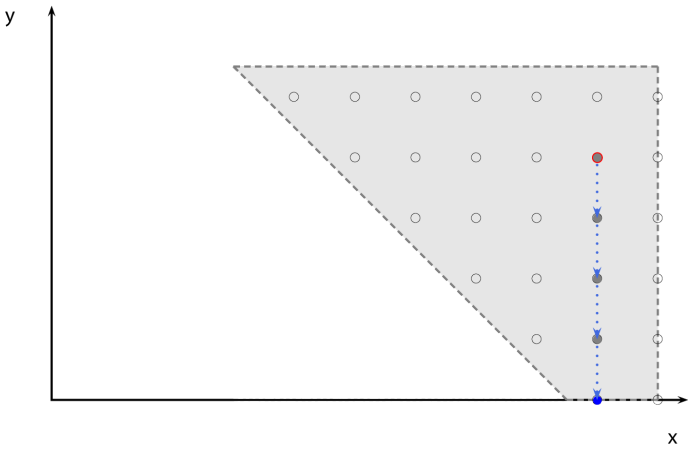


FIGURE 3.4. Partial reduction steps of $(9,4,9,3,1)$ by $\mathcal{G}_1$

Then, reducing each remainder by $\mathcal{G}_2$*:*

$pRem((9,0,1,11,1),\mathcal{G}_2) = \{(9,0,1,11,1),\ (8,1,1,9,2),\ (7,2,1,9,3),\ (6,3,1,5,4),\ (5,4,1,3,5),\ (4,5,1,1,6)\}$.

The movements to obtain those remainders from $(9,0,1,11,1)$ *in the* x*-*y*-plane are described in Figure 3.5. The blue dots are those final remainders (the nondominated solutions), the grey dots are those that have been discarded by the domination criterion and the ones with a red circumference is the initial point for this reduction.*

The entire set of Pareto-optimal solutions is:

$$\{(9,0,1,11,1),(8,1,1,9,2),(7,2,1,7,3),(6,3,1,5,4),(5,4,1,3,5),(4,5,1,1,6)\}$$

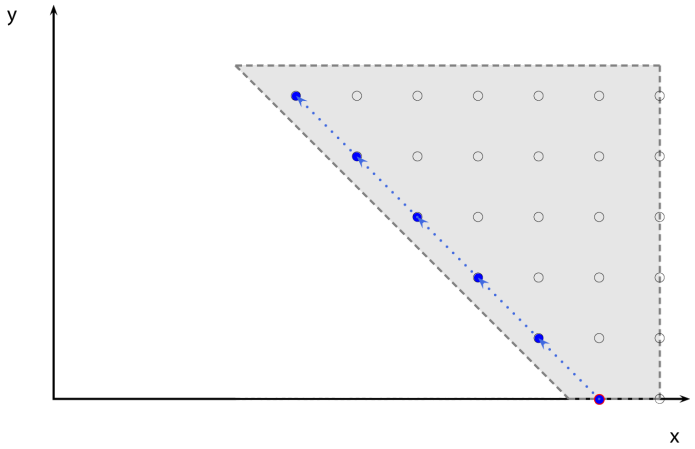


FIGURE 3.5. Partial reduction steps of $pRem((9,4,9,3,1),\mathcal{G}_1)$ by $\mathcal{G}_2$

If instead of considering first $\mathcal{G}_1$ and then $\mathcal{G}_2$, we could do it first for $\mathcal{G}_2$ and then for $\mathcal{G}_1$, obtaining the same solutions. The illustrations with this alternative movements is shown in the following graphs:

Figure 3.7 shows the feasible region and the Pareto-optimal solutions of the example above.

EXAMPLE 3.2.2.

$$\begin{array}{lll} \min & \{-2x+y, x-2y\} & \\ s.t. & & \\ & 4x-7y & \leqslant 5 \\ & x+8y & \leqslant 50 \\ & 3x-y & \geqslant 0 \\ & x+y & \geqslant 4 \\ & x,y & \in \mathbb{Z}_+ \end{array} \tag{15}$$

The feasible region and the improvement cone for this problem is shown in Figure 3.8. The geometric Gröbner basis for this problem has 4 maximal chains defining 14 different movements. The Hasse diagram for the x-y projection of the p-Gröbner basis is shown in Figure 3.9.

From the initial solution $(3,4)$, Figure 3.10 shows the movements from that point to the nondominated solutions:

$$\{(10,5),(9,5),(8,5),(7,5),(6,5),(5,5),(4,5),(2,6)\}.$$

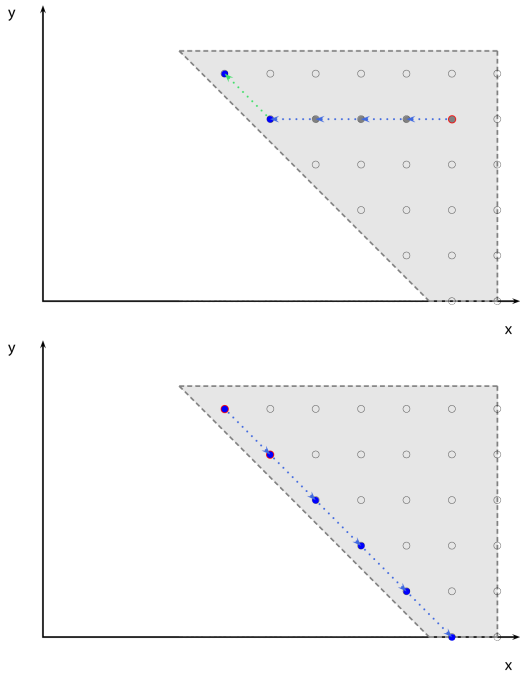


FIGURE 3.6. Two ways to compute remainders by the basis in Example 3.2.1.

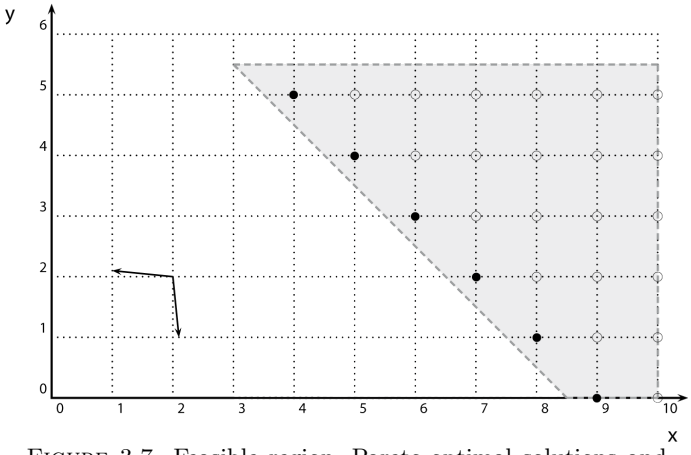


FIGURE 3.7. Feasible region, Pareto-optimal solutions and improvement cone for Example 3.2.1

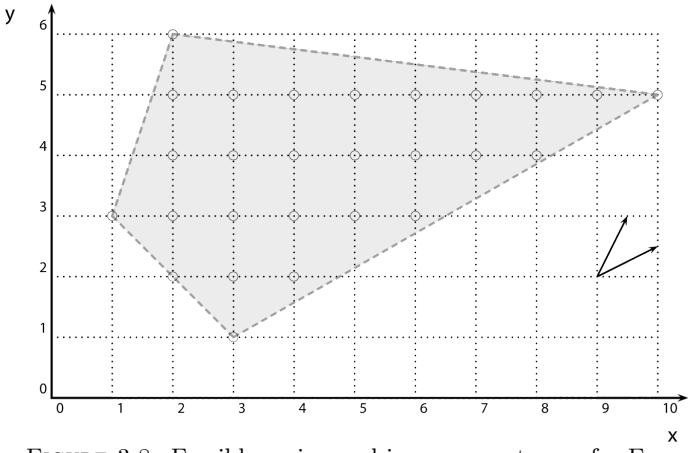


FIGURE 3.8. Feasible region and improvement cone for Example 3.2.2.

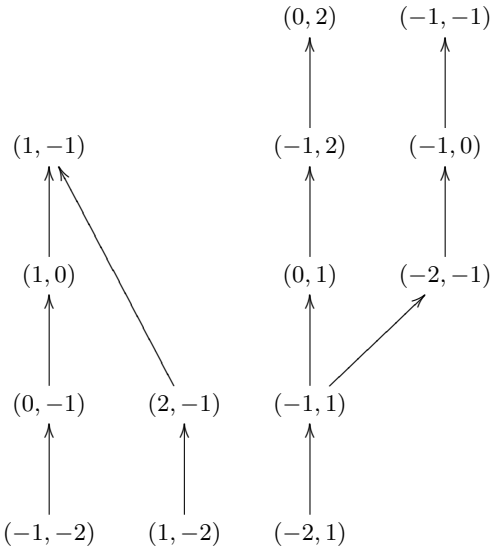


FIGURE 3.9. Hasse diagram for of the elements in the p-Gröbner basis for Example 3.2.2.

3.4. Computational Experiments

A series of computational experiments have been performed in order to evaluate the behavior of the proposed solution method. Programs have been coded in MAPLE 10 and executed in a PC with an Intel Pentium 4 processor at 2.66GHz and 1 GB of RAM. In the implementation of Algorithm 11 to

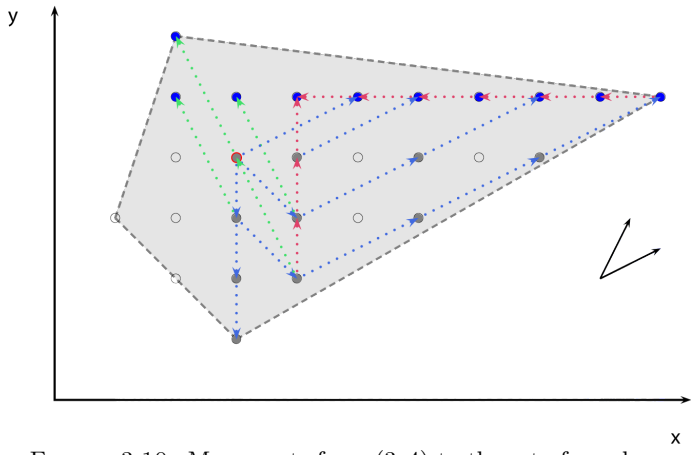


FIGURE 3.10. Movements from $(3, 4)$ to the set of nondominated solutions.

obtain the p-Gröbner basis, the package *poset* for Maple [96] has been used to compute, at each iteration, the maximal chains for the p-Gröbner basis. The implementation has been done in a symbolic programming language, available upon request, in order to make the access easy to both optimizers and algebraic geometers.

The performance of the algorithm was tested on randomly generated instances for knapsack and transportation [82] multiobjective problems for 2, 3 and 4 objectives. For the knapsack problems, 4, 5 and 6 variables programs have been considered, and for each group, the coefficients of the constraint were randomly generated in $[0, 20]$ whereas the coefficients of the objective matrices range in $[0, 20]$. Once the constraint vector, $(a_1, \ldots, a_n)$, is generated, the right-hand side is fixed as $b = \lceil \frac{1}{2} \sum_{i=1}^{n} a_i \rceil$ to ensure feasibility.

The computational tests for each number of variables have been done in the following way: (1) Generate 5 constraint vectors and compute the initial system of generators for each of them using Algorithm 7; (2) Generate five random objective matrices for each number of objectives (2, 3 and 4) and compute the corresponding p-Gröbner basis using Algorithm 11; and (3) with $b = \lceil \frac{1}{2} \sum_{i=1}^{n} a_i \rceil$ and for each objective matrix, compute the Pareto-optimal solutions using Algorithm 12.

Table 4.1 contains a summary of the average results obtained for the considered knapsack multiobjective problems. The second, third and fourth columns show the average CPU times for each stage in the algorithm: `sogt` is the CPU time for computing the system of generators, `pgbt` is the CPU time for computing a p-Gröbner basis, and `post` is the time for computing a feasible

solution and partially reducing it to obtain the set of Pareto-optimal solutions. The fifth column shows the total time for computing the set of Pareto-optimal solutions for the problem. Finally, the sixth and seventh columns show the average number of Pareto-optimal solutions and the number of maximal chains in the p-Gröbner basis for the problem. The problems have been named as `knapN_O` where `N` is the number of variables and `O` is the number of objectives. For the transportation problems, instances with 3 origins × 2 destinations, 3

problem	sogt	pgbt	post	tott	|pos|	|maxch|	steps	act_pGB
knap4_2	0.063	249.369	1.265	250.697	11	20	2	164.920
knap4_3	0.063	1002.689	2.012	1004. 704	5	46	2	772.772
knap4_4	0.063	1148.574	2.374	1151.011	16	98	2.4	763.686
knap5_2	0.125	1608.892	0.875	609.892	3	29	2	1187.201
knap5_3	0.125	3500.831	2.035	3503.963	2	30	2.2	2204.123
knap5_4	0.125	3956.534	2.114	3958.773	9	45.4	3	3044.157
knap6_2	0.185	2780.856	2.124	2783.165	18	156	2.4	2241.091
knap6_3	0.185	3869.156	2.018	3871.359	16.4	189	2.4	2790.822
knap6_4	0.185	4598.258	3.006	4601.449	26	298	3.2	3096.466

TABLE 3.1. Summary of computational experiments for knapsack problems

origins × 3 destinations and 4 origins × 2 destinations have been considered. In this case, for each fixed numbers of origins, s, and destinations, d, the constraint matrix, $A \in Z^{(s+d)\times(sd)}$, is fixed. Then, we have generated 5 instances for each problem of size $s \times d$. Each of these instances is combined with 5 different right-hand side vectors. The procedure is analogous to the knapsack computational test: a first step where a system of generators is computed, a second one, where the p-Gröbner basis is built and in the last step, the set of Pareto-optimal solutions is computed using partial reductions. Table 3.2 shows the average CPU times and the average number of Pareto-optimal solutions and maximal chains in the p-Gröbner basis for each problem. The `steps` column shows the average number of steps in the p-Gröbner computation, and `act_pGB` is the average CPU time in the computation of the p-Gröbner basis elapsed since the last element was added to the basis until the end of the process. The problems have been named as `trNxM_O` where `N` is the number of origins, `M` is the number of destinations and `O` is the number of objectives. As can be seen in tables 4.1 and 3.2, the overall CPU times are clearly divided into the three steps, being the most costly the computation of the p-Gröbner basis. In all the cases more than 99% of the total time is spent computing the p-Gröbner basis. Once this structure is computed, obtaining the Pareto-optimal solutions is done very efficiently.

problem	sogt	pgbt	post	tott	|pos|	|maxch|	steps	act_pGB
tr3x2_2	0.015	11.813	0.000	11.828	5.2	6	2	7.547
tr3x2_3	0.015	7.218	13.108	30.341	12	2.6	2	6.207
tr3x2_4	0.015	6.708	15.791	21.931	6	5	2.2	4.561
tr3x3_2	0.047	1545.916	1.718	1547.681	5	92	2	928.222
tr3x3_3	0.047	3194.333	11.235	3205.615	9	122	2.4	2172.146
tr3x3_4	0.047	3724.657	7.823	3732.527	24	187.4	2.2	2112.287
tr4x2_2	0.046	675.138	2.122	677.306	3.4	35.2	2	398.093
tr4x2_3	0.046	1499.294	6.288	1505.628	5.8	42.4	2.2	119.519
tr4x2_4	0.046	2285.365	7.025	2292.436	12	59	2.2	1654.048

TABLE 3.2. Summary of computational experiments for the battery of multiobjective transportation problems

The CPU times and sizes in the different steps of the algorithm are highly sensitive to the number of variables. However, our algorithm is not very sensitive to the number of objectives, since the increment of CPU times with respect to the number of objectives is much smaller than the one with respect to the number of variables.

It is clear that one can not expect fast algorithms for solving MOILP, since all these problems are NP-hard. Nevertheless, our approach provides exact tools that apart from solving these problems, give insights into the geometric and algebraic nature of the problem.

As mentioned above, using our methodology one can identify the common algebraic structure within any multiobjective integer linear problem. This connection allows to improve the efficiency of our algorithm making use of any advance that improves the computation of Gröbner bases. In fact, any improvements of the standard Gröbner bases theory may have an impact in improving the performance of this algorithm. In particular, one can expect improvements in the efficiency of our algorithm based on the special structure of the integer program (see for instance Remark 3.2.1). In addition, we have to mention another important issue in our methodology. As shown in Theorem 3.2.2, solving MOILP with the same constraint and objective matrices requires computing only once the p-Gröbner basis. Therefore, once this is done, we can solve different instances varying the right-hand side very quickly.

Finally, we have observed from our computational tests that a significant amount of the time, more than 60% (see column `act_pGB`), for the computation of the p-Gröbner basis is spent checking that no new elements are needed in this structure. This implies that the actual p-Gröbner basis is obtained much earlier than when the final test is finished. A different truncation strategy may be based on the number of steps required to obtain the p-Gröbner basis. According to the exact method, the algorithm stops once in a step no new

elements are added to the structure. Our tables show that in most cases the number of steps is 2, actually only one step is required to generate the entire p-Gröbner basis (see column `steps`). These facts can be used to accelerate the computational times at the price of obtaining only heuristic Pareto-optimal solutions. This idea may be considered an alternative primal heuristic in MOILP and will be the subject of further research.

CHAPTER 4

Short generating functions

This chapter addresses another tool for solving MOILP: generating functions. Here, we present two algorithms for solving general multiobjective problems: (1) fixing the dimension of the decision space, a polynomial time algorithm that encodes the set of nondominated solutions of the problem as a short sum of rational functions; and (2) a digging algorithm that computes the entire set of nondominated solutions using the multivariate Laurent expansion for the Barvinok's function of the polytope defined by the constraints of the problem. Furthermore, two polynomial delay algorithms for solving multiobjective problems are also presented (fixing only the dimension of the decision space).

We show the results of a computational experiment and its analysis. Here, we solve biobjective knapsack problems, report on the performance of the algorithms and draw some conclusions on their results and their implications.

At the end of the chapter, we apply short generating functions to an algebraic problem: counting the number of numerical semigroups of a given genus. We translate that problem into the problem of determining the number of integer points inside a convex polyhedron. Finally we show the results of counting the number of numerical semigroups with genuses up to 15.

4.1. The multiobjective problem

Through this chapter, we deal with maximization multiobjective linear and integer problems of the form:

$$
(MIP^{*}_{A,C}(b)) \qquad
\begin{array}{lll}
max & (c_1\, x, \ldots, c_k\, x) & \\
s.t. & & \\
& \displaystyle\sum_{j=1}^{d} a_{ij}\, x_j \leq b_i & i = 1, \ldots, m \\
& x_j \in \mathbb{Z}_{+} & j = 1, \ldots, n
\end{array}
$$

with a_{ij}, b_i integers and x_i non negative. As in the previous chapters, we will consider the above problem in its standard form, i.e., the coefficient of the k objective functions are non-negative and the constraints are in equation form.

In addition, we will assume that the constraints define a polytope (bounded) in $\mathbb{R}^n$. Therefore, from now on we deal with $MIP^*_{A,C}(b)$ (the superscript * denotes that the problem is in maximization form).

In this case, a vector $\widehat{x} \in \mathbb{R}^n$ is said to be a *nondominated* solution of $MIP^*_{A,C}(b)$ if there is no other feasible vector y such that

$$c_j\, y \geq c_j\, \widehat{x} \qquad \forall j = 1, \ldots, k$$

with at least one strict inequality for some j. If x is a nondominated solution, the vector $Cx = (c_1\, x, \ldots, c_k\, x) \in \mathbb{R}^k$ is called *efficient*. Recall that X_E is a subset of $\mathbb{R}^n$ (*decision space*) and Y_E is a subset of $\mathbb{R}^k$ (*objectives space*). A dominated point, y, is dominated by x if $c_i\, x \gneqq c_i\, y$ for all $i = 1, \ldots, k$.[1] We denote by X_E the set of all nondominated solutions for ($MIP^*_{A,C}(b)$) and by Y_E the image under the objective functions of X_E, that is, $Y_E = \{C\, x : x \in X_E\}$.

4.2. A short rational function expression of the entire set of nondominated solutions

We present in this section an algorithm for solving $MIP^*_{A,C}(b)$ using Barvinok's rational functions technique.

THEOREM 4.2.1. *Let $A \in \mathbb{Z}^{m\times n}$, $b \in \mathbb{Z}^m$, $C = (c_1, \ldots, c_k) \in \mathbb{Z}^{k\times n}$, $J \in \{1, \ldots, n\}$, and assume that the number of variables n is fixed. Suppose $P = \{x \in \mathbb{R}^n : A\, x \leq b, x \geq 0\}$ is a rational convex polytope in $\mathbb{R}^n$. Then, we can encode, in polynomial time, the entire set of nondominated solutions for $MIP^*_{A,C}(b)$ in a short sum of rational functions.*

PROOF. Using Barvinok's algorithm (Theorem 1.4.1 - Theorem 5.4 in [7]), we compute the following generating function in $2n$ variables:

$$f(x, y) := \sum_{(u,v)\in P_C\cap\mathbb{Z}^{2n}} x^u\, y^v \tag{16}$$

where $P_C = \{(u, v) \in \mathbb{Z}^n \times \mathbb{Z}^n : u, v \in P, c_i\, u - c_i\, v \geq 0 \text{ for all } i = 1, \ldots, k \text{ and } \sum_{i=1}^{k} c_i\, u - \sum_{i=1}^{k} c_i\, v \geq 1\}$. P_C is clearly a rational polytope. For fixed $u \in \mathbb{Z}^n$, the y-degrees, α, in the monomial $x^u\, y^\alpha$ of $f(x, y)$ represent the solutions dominated by u.

Now, for any function φ, let $\pi_{1,\varphi}, \pi_{2,\varphi}$ be the projections of $\varphi(x, y)$ onto the x- and y-variables, respectively. Thus $\pi_{2,f}(y)$ encodes all dominated feasible integral vectors (because the degree vectors of the x-variables dominate them, by construction), and it can be computed from $f(x, y)$ in polynomial time by Theorem 1.7 in [7].

[1] We are denoting by $\gneqq$ the binary relation "greater than or equal to" and where it is assumed that at least one of the inequalities in the list is strict.

Let $V(P)$ be the set of extreme points of the polytope P and choose an integer $R \geq \max\{v_i : v \in V(P), i = 1, \ldots, n\}$ (we can find such an integer R via linear programming). For this positive integer, R, let $r(x, R)$ be the rational function for the polytope $\{u \in \mathbb{R}^n_+ : u_i \leq R\}$, its expression is:

$$r(x, R) = \prod_{i=1}^{n} \left(\frac{1}{1 - x_i} + \frac{x_i^R}{1 - x_i^{-1}} \right).$$

Define $f(x, y)$ as above, $\pi_{2,f}(x)$ the projection of f onto the second set of variables as a function of the x-variables and $F(x)$ the short generating function of P. They are computed in polynomial time by Theorem 1.7 and Theorem 5.4 in [7] respectively. Compute the following difference:

$$h(x) := F(x) - \pi_{2,f}(x).$$

This is the sum over all monomials x^u where $u \in P$ is a nondominated solution, since we are deleting, from the total sum of feasible solutions, the set of dominated ones.

This construction gives us a short rational function associated with the sum over all monomials with degrees being the nondominated solutions for $MIP^*_{A,C}(b)$. This function encodes the whole set of nondominated solutions. As a consequence, we can also compute the number of nondominated solutions for the problem. The complexity of the entire construction being polynomial since we only use polynomial time operations among four short rational functions of polytopes (these operations are the computation of the short rational expressions for $f(x, y)$, $F(x)$, $r(x, R)$ and $\pi_{2,f}(x)$). □

REMARK 4.2.1. *To prove the above result one may use a different approach to compute the nondominated solutions assuming that there exists a polynomially bounded (for fixed dimension) feasible lower bound set, L, for* $MIP^*_{A,C}(b)$*, i.e., a set of feasible solutions such that every nondominated solution is either one element in L, or it dominates at least one of the elements in L. The reader may note that this assumption is rather useful and in fact, it easily applies to most combinatorial optimization problems in maximization form with nonnegative objective matrices. For instance, knapsack, transportation, assignment or matching problems, among many others.*

First, compute the following operations with generating functions:

$$H(x, y) = f(x, y) - f(x, y) * (\pi_{2,f}(x)\, r(y, R))$$

where $*$ stands for the Hadamard product[2].

This is the sum over all monomials $x^u\, y^v$ where $u, v \in P$, u is a nondominated solution and v is dominated by u. In $H(x,y)$, each nondominated solution, u, appears as many times as the number of feasible solutions that it dominates.

Next, compute a feasible lower bound set (see [52; 47]), $L = \{\alpha_1, \ldots, \alpha_s\}$. This way the set of nondominated solutions is encoded using the following construction:

Let $RLB^i(x,y)$ be the following short sum of rational functions

$$RLB^i(x,y) = H(x,y) * (y^{\alpha_i}\, r(x,R)) \qquad i = 1, \ldots, s.$$

Taking into account that for each i, the element y^{α_i} is common factor for $RLB^i(x,y)$ and it is the unique factor where the y-variables appear, we can define $ND^i(x) = \dfrac{RLB^i(x,y)}{y^{\alpha_i}}, i = 1, \ldots, s$, to be the sum of rational functions that encodes the nondominated solutions that dominate α_i, $i = 1, \ldots, s$. Therefore, the entire set of nondominated solutions for $MIP^*_{A,C}(b)$ is encoded in the short sum of rational functions $ND(x) = \sum_{i=1}^{k} ND^i(x)$.

4.3. Digging algorithm for the set of nondominated solutions of MOILP

Section 4.2 proves that encoding the entire set of efficient solutions of MOILP can be done in polynomial time for fixed dimension. This is a compact representation of the solution concept. Nevertheless, one may be interested in an explicit description of this list of points. This task could be performed, by expanding the short rational expression which is ensured by Theorem 4.2.1, but it would require the implementation of all operations used in the proof. As far we know, they have never been efficiently implemented.

An alternative algorithm for enumerating the nondominated solutions of a multiobjective integer programming problem, which uses rational generating functions, is the digging algorithm. This algorithm is an extension of a heuristic proposed by Lasserre [71] for the single-objective case.

Let A, C and b be as in $MIP^*_{A,C}(b)$, and assume that $P = \{x \in \mathbb{R}^n : Ax \leq b, x \geq 0\}$ is a polytope. Then, by Theorem 1.4.1, we can compute a rational

[2]Given $g_1(z) = \sum_{m \in \mathbb{Z}^d} \beta_m\, z^m$ and $g_2(z) = \sum_{m \in \mathbb{Z}^d} \gamma_m\, z^m$, the Hadamard product $g = g_1 * g_2$ is the power series $g(z) = \sum_{m \in \mathbb{Z}^n} \eta_m\, z^m$ where $\eta_m = \beta_m \gamma_m$).

expression for $f(P;z) = \sum_{\alpha \in P \cap \mathbb{Z}^n} z^\alpha$ in the form

$$f(P;z) = \sum_{i \in I} \varepsilon_i \frac{z^{u_i}}{\prod_{j=1}^{n} (1 - z^{v_{ij}})}$$

in polynomial time for fixed dimension, n. Each addend in the above sum will be referred to as f_i, $i \in I$.

If we make the substitution $z_i = z_i\, t_1^{c_{1i}} \cdots t_k^{c_{ki}}$, in the monomial description we have $f(P;z,t_1,\ldots,t_k) = \sum_{\alpha \in P \cap \mathbb{Z}^n} z^\alpha\, t_1^{c_1 \alpha} \cdots t_k^{c_k \alpha}$, where $c_1,\ldots,c_k$ are the rows in C. It is clear that for enumerating the entire set of nondominated solutions, it would suffice to look for the set of leading terms, in the t-variables, in the partial order induced by C, $\succ_C$, of the multi-polynomial $f(P;z,t_1,\ldots,t_k)$. After the above changes we have:

$$f(P;z,t_1,\ldots,t_k) = \sum_{i \in I} f_i(P;z,t_1,\ldots,t_k), \tag{17}$$

where $f_i(P;z,t_1,\ldots,t_k) := \varepsilon_i \dfrac{z^{u_i}\, t_1^{c_1 u_i} \cdots t_k^{c_k u_i}}{\prod_{j=1}^{n} (1 - z^{v_{ij}}\, t_1^{c_1 v_{ij}} \cdots t_k^{c_k v_{ij}})}$. Now, we can assume, wlog, that $c_1\, v_{ij}$ is negative or zero. If it were zero, then we could assume that $c_2\, v_{ij}$ is negative. Otherwise, we would repeat the argument until the first non zero element is found (it is assured that this element exists, otherwise the factor would not appear in the expression of the short rational function). Indeed, if the first non zero element were positive, we would make the change:

$$\frac{1}{1 - z^{v_{ij}}\, t_1^{c_1 v_{ij}} \cdots t_k^{c_k v_{ij}}} = \frac{-z^{-v_{ij}}\, t_1^{-c_1 v_{ij}} \cdots t_k^{-c_k v_{ij}}}{1 - z^{-v_{ij}}\, t_1^{-c_1 v_{ij}} \cdots t_k^{-c_k v_{ij}}}$$

and the sign of the t_1-degree would be negative.

With these assumptions, the multivariate Laurent series expansion for each rational function, f_i, in $f(P;z,t_1,\ldots,t_k)$ is

$$\begin{aligned} f_i &= \varepsilon_i z^{u_i}\, t_1^{c_1 u_i} \cdots t_k^{c_k u_i} \prod_{j=1}^{d} \sum_{\lambda=0}^{\infty} t_1^{\lambda c_1 v_{ij}} \cdots t_k^{\lambda c_k v_{ij}} \\ &= \varepsilon_i z^{u_i}\, t_1^{c_1 u_i} \cdots t_k^{c_k u_i} \prod_{j=1}^{d} (1 + t_1^{c_1 v_{ij}} \cdots t_k^{c_k v_{ij}} + t_1^{2c_1 v_{ij}} \cdots t_k^{2c_k v_{ij}} + \cdots) \end{aligned}$$

The following result allows us to develop a finite algorithm for solving $MIP^*_{A,C}(b)$ using Barvinok's rational generating functions.

Let U (resp. l) be the greatest (resp. smallest) value that appears in the non-zero absolute values of the entries in A, b, C. Set $M = \max\{U, l^{-1}\}$.

LEMMA 4.3.1. *Obtaining the entire set of nondominated solutions for a MOILP requires only an explicit finite, polynomially bounded (in M) number of terms of the long sum in the Laurent expansion of $f(P; z, t_1, \ldots, t_k)$.*

PROOF. Let $i \in I$, $j \in \{1, \ldots, n\}$ and define $P_i = \{\lambda \in \mathbb{Z}_+^n : c_s u_i + \sum_{r=1}^{n} \lambda_r c_s v_{ir} \geq 0, s = 1, \ldots, k\}$, $M_{ij} = \max\{\lambda_j : \lambda \in P_i\}$ and $m_{ij} = \min\{\lambda_j : \lambda \in P_i\}$. M_{ij} and m_{ij} are well-defined because P_i, defined above, is non empty and bounded since, by construction, for each $j \in \{1, \ldots, n\}$ there exists $s \in \{1, \ldots, k\}$ such that $c_s v_{ij} < 0$.

Then, it is enough to search for the nondominated solutions in the finite sum

$$\varepsilon_i z^{u_i} t_1^{c_1 u_i} \cdots t_k^{c_k u_i} \prod_{j=1}^{d} \sum_{\lambda=m_{ij}}^{M_{ij}} t_1^{\lambda c_1 v_{ij}} \cdots t_k^{\lambda c_k v_{ij}}.$$

First, $m_{ij} \geq 0$. Then, by applying Cramer's rule one can see that M_{ij} is bounded above by $O(M^{2n+1})$. Thus, the explicit number of terms in the expansion of f_i, namely $\prod_{j=1}^{n} \lfloor M_{ij} - m_{ij} \rfloor$, is polynomial, when the dimension, n is fixed. □

The digging algorithm looks for the leading terms in the t-variables, with respect to the partial order induced by C. At each rational function (addends in the above sum (17)) multiplications are done in lexicographical order in their respective bounded hypercubes. If the t-degree of a specific multiplication is not dominated by one of the previous factors, it is kept in a list; otherwise the algorithm continues augmenting lexicographically the lambdas. To simplify the search at each addend, the following consideration can be taken into account: if $t_1^{\alpha_o + \sum_j \lambda_j \alpha_j^1} \cdots t_k^{\alpha_o + \sum_j \lambda_j \alpha_j^k}$ is dominated, then any term of the form $t_1^{\alpha_o + \sum_j \mu_j \alpha_j^1} \cdots t_k^{\alpha_o + \sum_j \mu_j \alpha_j^k}$, μ being componentwise larger than λ, is dominated as well.

The above process is done on each rational function that appears in the representation of f. As an output we get a set of leading terms (for each rational function), that are the candidates to be nondominated solutions. Terms that appear with opposite signs will be cancelled. Removing terms in the list of candidates (to be nondominated solutions) implies consideration of those terms that were dominated by the cancelled ones. These terms are included in the current list of candidates and the process continues until no more terms are added.

At the end, some dominated elements may appear in the union of the final list. Deleting them in a simple cleaning process gives the list that contains only the entire set of nondominated solutions for the multiobjective problem.

Algorithm 13 details the pseudocode of the digging algorithm.

Algorithm 13: Digging algorithm for multiobjective problems

input : $A \in \mathbb{Z}^{m\times n}$, $b \in \mathbb{Z}^m$, $C \in \mathbb{Z}^{k\times n}$

Step 1: (Initialization)

Compute, $f(z)$, the short sum of rational functions encoding the set of nondominated solutions of $MIP^*_{A,C}(b)$. The number of rational function is indexed by I.

Make the substitution $z_i = z_i\, t_1^{c_{1i}} \cdots t_k^{c_{ki}}$ in $f(z)$. Denote by f_i, $i \in I$, each one of the addends in f, as in (17).

Set m_{ij} and M_{ij}, $j = 1, \ldots, n$, the lower and upper bounds computed in the proof of Lemma 4.3.1 and $\mathcal{S} = \prod_{j=1}^{n} [m_{ij}, M_{ij}] \cap \mathbb{Z}^n_+$.

Set $\Gamma_i := \{\}$, $i \in I$, the initial set of nondominated solutions encoded in f_i.

Step 2: (Nondominance test)

repeat

  for $i \in I$ **do**

    for $\lambda^i \in \mathcal{S}$ *such that its entries are not componentwise larger than a previous* λ **do**

      Compute $p_i := z^{w_o}\, t_1^{w_1} \cdots t_k^{w_k}$, being $w_o := u_i + \sum_{j=1}^{n} \lambda^i_j\, v_{ij}$

      and $w_h := c_1\, u_i + \sum_{j=1}^{n} \lambda^i_j\, c_h\, v_{ij} \qquad h = 1, \ldots, k$

      if p *is nondominated by elements in* Γ_i **then**

        $\Gamma_i \leftarrow \Gamma_i \cup \{p\}$

      end

    end

  end

  Step 3: (Feasibility test)

  for $s, r \in I$, $s < r$ **do**

    if $p \in \Gamma_j \cap \Gamma_h$, $\varepsilon_j = -\varepsilon_h$ **then**

      $\Gamma_j \leftarrow \Gamma_j \setminus \{p\}$

      $\Gamma_h \leftarrow \Gamma_h \setminus \{p\}$

    end

  end

until *No changes in any* Γ_i *are done for all* $i \in I$;

$\Gamma := \bigcup_j \Gamma_j$.

Remove from Γ the dominated elements.

output: The entire set of nondominated solutions for $MIP^*_{A,C}(b)$: Γ

Taking into account Lemma 4.3.1 and the fact that Algorithm 13 never cycles, we have the following statement.

Recall that $M = \max\{U, l^{-1}\}$, where U is the greatest value that appears in the non-zero absolute values of the entries in A, b, C and l is the smallest value among these values.

THEOREM 4.3.1. *Algorithm 13 computes in a finite (polynomially bounded on M) number of steps, the entire set of nondominated solutions for $MIP^*_{A,C}(b)$.*

It is well known that enumerating the nondominated solutions of MOILP is NP-hard and #P-hard ([41; 46]). Thus, one cannot expect to have very efficient algorithms for solving the general problem (when the dimension is part of the input).

In the following, we concentrate on a different concept of complexity that has been already used in the literature for slightly different problems. Computing maximal independent sets on graphs is known to be #P-hard ([55]), nevertheless there exist algorithms for obtaining these sets which ensure that the number of operations necessary to obtain two consecutive solutions of the problem is bounded by a polynomial in the problem input size (see e.g. [106]). These algorithms are called polynomial delay. Formally, an algorithm is said *polynomial delay* if the delay, which is the maximum computation time between two consecutive outputs, is bounded by a polynomial in the input size ([4; 65]).

In our case, a polynomial delay algorithm, in fixed dimension, for solving a multiobjective linear integer program means that once the first nondominated solution is computed, either in polynomial time a next nondominated solution is found or the termination of the algorithm is given as an output.

Next, we present a polynomial delay algorithm, in fixed dimension, for solving multiobjective integer linear programming problems. This algorithm combines the theoretical construction of Theorem 4.2.1 and a digging process in the Laurent expansion of the short rational functions of the polytope associated with the constraints of the problem.

The algorithm proceeds as follows.

Let $f(z)$ be the short rational function that encodes the nondominated solutions (by Theorem 4.2.1, the complexity of computing f is polynomial -in fixed dimension-). Make the changes $z_i = z_i\, t_1^{c_{1i}} \cdots t_k^{c_{ki}}$, for $i \in I$, in f. Denote by f_i each of the rational functions of f after the above changes. Next, the Laurent expansion over each rational function, f_i, is done in the following way: (1) Check if f_i contains nondominated solutions computing the Hadamard product of f_i with f. If f_i does not contain nondominated solutions, discard it and set $I := I \backslash \{i\}$ (termination); (2) if f_i encodes nondominated

solutions, look for an arbitrary nondominated solution (expanding f_i); (3) once the first nondominated solution, α, is found, check if there exist more nondominated solutions encoded in the same rational function computing $f * (f_i - z^{\alpha}\, t_1^{c_1\alpha} \cdots t_k^{c_k\alpha})$. If there are more solutions encoded in f_i, look for them in $f_i - z^{\alpha}\, t_1^{c_1\alpha} \cdots t_k^{c_k\alpha}$. Repeat this process until no new nondominated solutions can be found in f_i.

The process above describes the pseudocode written in Algorithm 14.

Algorithm 14: A polynomial delay algorithm for solving MOILP

input : $A \in \mathbb{Z}^{m\times n}$, $b \in \mathbb{Z}^m$, $C \in \mathbb{Z}^{k\times n}$
output: The entire set of nondominated solutions for $MIP^*_{A,C}(b)$
Set $X_E = \{\}$ and $Y_E = \{\}$.
Step 1: Compute, $f(z)$, the short sum of rational functions encoding the set of nondominated solutions of $MIP^*_{A,C}(b)$. The number of rational function is indexed by I.
Make the substitution $z_i = z_i\, t_1^{c_{1i}} \cdots t_k^{c_{ki}}$ in $f(z)$. Denote by f_i, $i \in I$, each one of the addends in f ($f = \sum_{i\in I} f_i$).
Step 2: For each $i \in I$, check $f_i * f$. If the set of lattice points encoded by this rational function is empty, do $I \leftarrow I \setminus \{i\}$.
Step 3:
while $I \neq \emptyset$ **do**
  for $i \in I$ **do**
    Look for the first nondominated solution, α, that appears in the Laurent expansion of f_i.
    Set $X_E \leftarrow X_E \cup \{\alpha\}$ and $Y_E \leftarrow Y_E \cup \{C\,\alpha\}$.
    Set $f_i \leftarrow f_i - z^{\alpha}\, t_1^{c_1\alpha} \cdots t_k^{c_k\alpha}$
    and check if $f * f_i$ encodes lattice points. If it does not encode lattice points, discard f_i ($I \leftarrow I \setminus \{i\}$) since f_i does not encode any other nondominated point, otherwise repeat.
  end
end

THEOREM 4.3.2. *Assume n is a constant. Algorithm 14 provides a polynomial delay (bounded on M) procedure to obtain the entire set of nondominated solutions of $MIP^*_{A,C}(b)$.*

PROOF. Let f be the rational function that encodes the nondominated solutions of $MIP^*_{A,C}(b)$. Theorem 4.2.1 ensures that f is a sum of short rational functions that can be computed in polynomial time.

Algorithm 14 digs separately on each one of the rational functions f_i, $i \in I$,that define f. (Recall that $f = \sum_{i\in I} f_i$).

Fix $i \in I$. First, the algorithm checks whether f_i encodes some nondominated solutions. This test is doable in polynomial time by Theorem 1.4.2.

If the answer is positive, an arbitrary nondominated solution is found among those encoded in f_i. This is done using digging and the Intersection Lemma. Specifically, the algorithm expands f_i on the hyperbox $\prod_{j=1}^{n}[m_{ij}, M_{ij}] \cap \mathbb{Z}^n$ and checks whether each term is nondominated. The expansion is polynomial, for fixed n, since the number of terms is polynomially bounded by Lemma 4.3.1. The test is performed using the Hadamard product of each term with f.

The process is clearly a polynomial delay algorithm. We use digging separately on each rational function f_i that encodes nondominated points. Thus, the time necessary to find a new nondominated solution from the last one is bounded by the application of digging on a particular f_i which, as argued above, is polynomially bounded. □

Instead of the above algorithm one can use a binary search procedure to solve multiobjective problem using short generating functions. In the worst case, in digging algorithm, the expansion of every nonnegative term is needed to obtain the set of nondominated solutions. Furthemore, as it is stated in Theorem 4.3.1, the number of steps to solve the problem is polynomially bounded on M. With a binary search approach, the number of steps to obtain the solutions of our problem is decreased to a number bounded on $log(M)$.

This alternative procedure can be described as follows:

Define M as above. By definition, $P \subseteq [0, M]^n$. The methodology is based on dividing the hypercube $[0, M]^n$ in 2^n parts, and repeat the process over those *sub-hypercubes* where obtaining a nondominated solution is assured, and until there is only one nondominated solution in this hypercube. For checking if there is a nondominated solution and counting if there is just one solution in a hypercube, we use Theorem 4.2.1 to encode the nondominated solutions in a short generating function, $h(x)$. It is not difficult to see that the short rational expression for the hypercube $\mathcal{H} = \prod_{i=1}^{n}[m_i, M_i] \subseteq \mathbb{R}^n$, with $m_i, M_i \in \mathbb{Q}$ for $i = 1, \ldots, n$, is:

$$r_{\mathcal{H}}(x) = \prod_{i=1}^{n} \left[\frac{x_i^{m_i}}{1 - x_i} + \frac{x_i^{M_i}}{1 - x_i^{-1}}\right]$$

Then, $h(x) * r_{\mathcal{H}}(x)$ encodes the subset of nondominated solutions that lie in the hypercube $\mathcal{H}$. One can count the number of lattice points encoded by this expression, i.e., the number of nondominated solutions in this hypercube. The algorithm proceeds as a depth first search. Once a hypercube, where a nondominated solution is found, and the hypercube is totally divided until the nondominated solution is located in a sub-hypercube with only this solution, the solution is kept, this sub-hypercube (node) is discarded for future checking

and the process is repeated for the next sub-hypercube. When every sub-hypercube in a hypercube is checked, discard it for future search and repeat for the predecessor hypercube, until there are no more predecessors.

An illustrative example of this procedure is shown in Figure 4.1 where it can be seen how the initial hypercube, $[0,4] \times [0,4]$, is divided successively in sub-hypercubes, until an isolated nondominated solution is located in one of them.

The finiteness of this procedure is assured since the number of times that the hypercube $[0,M]^n$ can be divided in 2^n sub-hypercubes is bounded by $log(M)$.

The pseudocode for this procedure is shown in Algorithm 15.

Algorithm 15: Binary algorithm for solving MOILP using SGF.

Initialization: $\mathcal{M} = [0,M]^n \subseteq P$.
Step 1: Let $\mathcal{M}_1, \ldots, \mathcal{M}_{2^n}$ be the hypercubes obtained dividing $\mathcal{M}$ by its central point.
$i = 1$
Step 2: **repeat**
  Count the elements encoded in $r_{\mathcal{M}_i}(x) * h(x)$: $n_{\mathcal{M}_i}$. This is the number of nondominated solutions in the hypercube $\mathcal{M}_i$.
  if $n_{\mathcal{M}_i} = 0$ **then**
    if $i < 2^n$ **then**
      $i \leftarrow i + 1$
    else
      Go to Step 1 with $\mathcal{M}$ the next hypercube to its predecessor hypercube
    end
  else
    if $n_{\mathcal{M}_i} = 1$ **then**
      Let x^* the unique solution in $\mathcal{M}_i$, $ND = ND \cup \{x^*\}$ and $i \leftarrow i + 1$
    else
      Go to Step 1 with $\mathcal{M} = \mathcal{M}_i$
    end
  end
until $i <= 2^n$;

THEOREM 4.3.3. *Assume n is a constant. Algorithm 14 provides a polynomial delay (bounded on $log(M)$) procedure to obtain the entire set of nondominated solutions of $MIP^*_{A,C}(b)$.*

REMARK 4.3.1. *The application of the above algorithm to the single criterion case provides an alternative proof of polynomiality for the problem of finding an optimal solution of integer linear problems, in fixed dimension.*

Assume that the number of objectives, k, is 1, and that there exists a unique optimal value for the problem. Compute the short sum of rational functions that encode this value. Locate the rational function, f_i, $i \in I$, which contains the optimum value. This operation takes polynomial time, for fixed n, computing the Hadamard product $f_i * f$, and counting the number of lattice points encoded by this function. Note that this is again doable in polynomial time using Barvinok's theory [9].

Once the rational function is found, expanding for finding the optimal value is computed in polynomial time, since by Lemma 4.3.1, it is enough to expand a prespecified polynomially bounded number of terms.

REMARK 4.3.2 (Optimization over the set of nondominated solutions). *In practice, a decision maker expects to be helped by the solutions of the multiobjective problem. In many cases, the set of nondominated solutions is too large to make easily the decision, so it is necessary to optimize (using a new criterion) over the set of nondominated solutions.*

With our approach, we are able to compute, in polynomial time for fixed dimension, a "short sum of rational functions"-representation, $F(z)$, of the set of nondominated solutions of $MIP^*_{A,C}(b)$. This representation allows us to re-optimize with a linear objective, ν, based in the algorithms for solving single-objective integer programming problems using Barvinok's functions: digging, binary search, ... or the algorithm proposed in Remark 5.1.1.

Moreover, the method of Remark 5.1.1 ensured polynomially of the algorithm in fixed dimension, as well as an explicit computation of the optimal values of the problem. The above discussion proves that solving the problem of optimizing a linear function over the efficient region of a multiobjective problem $MIP^*_{A,C}(b)$ is doable in polynomial time, for fixed dimension.

4.4. Computational Experiments

For illustrative purposes, a series of computational experiments have been performed in order to evaluate the behavior of a simple implementation of the digging algorithm (Algorithm 13). Computations of short rational functions have been done with Latte v1.2 [36] and Algorithm 13 has been coded in MAPLE 10 and executed in a PC with an Intel Pentium 4 processor at 2.66Gz and 1 GB of RAM. The implementation has been done in a symbolic programming language, available upon request, in order to make the access easy for the interested readers.

The performance of the algorithm was tested on randomly generated instances for biobjective (two objectives) knapsack problems. Problems from 4 to 8 variables were considered, and for each group, the coefficients of the

constraint were randomly generated in $[0, 20]$. The coefficients of the two objective matrices range in $[0, 20]$ and the coefficients of the right hand side were randomized in $[20, 50]$. Thus, the problems solved are in the form:

$$\max (c_1, c_2)\, x \quad s.t. \quad a_1 x_1 + \cdots + a_n\, x_n \leq b, x_i \in \mathbb{Z}_+ \tag{18}$$

The computational tests have been done on this way for each number of variables: (1) Generate 5 constraint vectors and right hand sides and compute the shorts rational functions for each of them; (2) Generate a random biobjective matrix and run digging algorithm for them to obtain the set of nondominated solutions.

Table 4.1 contains a summary of the average results obtained for the considered knapsack multiobjective problems. The second and third columns show the average CPU times for each stage in the Algorithm: `srf` is the CPU time for computing the short rational function expression for the polytope with LattE and `mo-digging` the CPU time for running the multiobjective digging algorithm for the problem. The total average CPU times are summarized in the `total` column. Columns `latpoints` and `nosrf` represent the number of lattice points in the polytope and the number of short rational functions, respectively. The average number of efficient solutions that appear for the problem is presented under `effic`. The problems have been named as `knapN` where `N` is the number of variables of the biobjective knapsack problem.

problem	srf	latpoints	nosrf	mo-digging	effic	total
knap4	0.018	12.25	25.75	4.863	4.5	4.881
knap5	0.038	31	62.5	487.640	9.25	487.678
knap6	0.098	217.666	124.25	2364.391	7.666	2364.489
knap7	0.216	325	203	2869.268	20	2869.484
knap8	0.412	3478	342	10245.533	46	10245.933

TABLE 4.1. Summary of computational experiments for knapsack problems

As can be seen in Table 4.1, the computation times are clearly divided into two steps (`srf` and `mo-digging`), being the most expensive the application of the digging algorithm (Algorithm 13). In all cases more than 99% of the total time is spent expanding the short rational function using "digging algorithm".

The CPU times and sizes in the two steps are highly sensitive to the number of variables. It is clear that one cannot expect fast algorithm for solving MOILP, since all these problems are NP-hard and #P-hard. Nevertheless, this approach gives exact tools for solving any MOILP problem, independently of the combinatorial nature of the problem.

Finally, from our computational experiments, we have detected that an easy, promising heuristic algorithm could be obtained truncating the expansion at each rational function. That algorithm would accelerate the computational times at the price of obtaining only heuristics nondominated points.

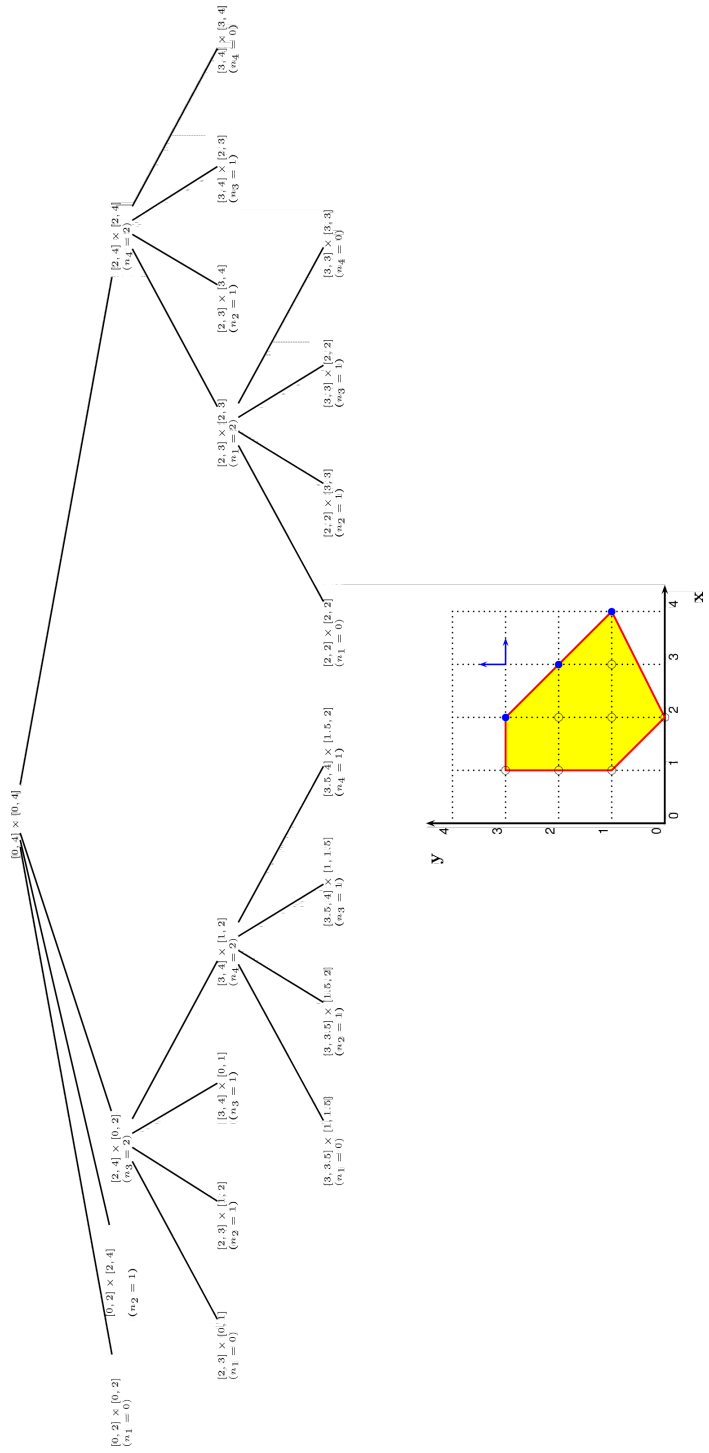


FIGURE 4.1. Search tree and feasible region for the problem $v - \max\{(x, y) : x + y \leq 5, x - 2y \leq 2, x + y \geq 2, x \geq 1, y \leq 3, x, y \in \mathbb{Z}_+\}$

4.5. Counting numerical semigroups of given genus

In this last section we apply generating functions to solve a problem apparently far from optimization: the one of counting the number of numerical semigroups with given genus.

A numerical semigroup is a subset S of $\mathbb{N}$ that is closed under addition, $0 \in S$ and generates $\mathbb{Z}$ as a group. This last condition is equivalent to $gcd(S) = 1$.

For given numerical semigroup S, the set $\mathbb{N} \setminus S$ has finitely many elements. Its maximum is known as the *Frobenius number* of S and it is denoted by $F(S)$. If $S = \mathbb{N}$, then $F(S) = -1$. Furthemore, S has a unique minimal system of generators $\{n_1, \ldots, n_p\}$. The element n_1 is the least positive integer belonging to S and it is denoted by $m(S)$, the *multiplicity of* S ($m(S) = min(S \setminus \{0\})$), and the set of elements in $G(S) = \mathbb{N} \setminus S$ is known as the set of *gap* of S. Its cardinality is known as the *genus* of S. The interested reader is referred to [6] or [88; 89] for further details.

Given $n \in S \setminus \{0\}$, the Apéry set (named so after [3]) of S with respect to n is the set $Ap(S, n) = \{s \in S : s - n \notin S\}$ and it can be easily shown that if for every $i \in \{0, \ldots, n-1\}$ we take $w(i)$ the least element in S congruent with i modulo n (denoted $w(i) \equiv i(mod n)$), then $Ap(S, n) = \{0 = w(0), w(1), \ldots, w(n-1)\}$. The set $Ap(S, n)$ completely determines S, since $S = \langle Ap(S, n) \cup \{n\} \rangle$ (where $\langle A \rangle$ denotes the monoid generated by A). Moreover, the set $Ap(S, n)$ contains in general more information than an arbitrary system of generators of S. For instance, $g(S) = max(Ap(S, n)) - n$ and for all $s \in S$ there exist unique $t \in \mathbb{N}$ and $w \in Ap(S, n)$ such that $s = tn + w$. One could say that the best way of describing S is by the Apéry set of one of its elements, and the smallest Apéry set is $Ap(S, m(S))$.

Through this section, we apply short generating functions for counting the number of solutions of certain system of diophantine equations that classify the numerical semigroups for fixed multiplicity and genus.

The use of generating functions is justified since once the short generating function is computed for a polytope $P \subseteq \mathbb{R}^n$, the number of lattice points inside P can be computed using tools from complex analysis. In the long expression, there are as many integral points in P as monomials, and then, if we evaluate the function in $z = 1$, we obtain that number. Using the short rational sum expression for P, the point $z = 1$ is always a pole, so, this expression cannot be evaluated at this point, but the limit in this point gives the same number.

First, we recall some results on numerical semigroups in order to translate the problem of finding the number of numerical semigroups for fixed genus,

in the problem of counting the number of lattice points inside a polytope and then, apply Barvinok's results.

Let $S(m)$ be the set of all numerical semigroups with multiplicity $m \in \mathbb{N}\setminus\{0\}$. In [90] Rosales et al. proved that there is a one-to-one correspondence between this set and the set of non-negative integer solutions of a system of linear Diophantine inequalities. This identification was previously used by Kunz in [70].

Let m be an integer greater than 1 and let S be in $\mathcal{S}(m)$ with $Ap(S,m) = \{0 = w(0), w(1), \ldots, w(m-1)\}$. For all $i \in \{1,\ldots,m-1\}$ let $k_i \in \mathbb{N}$ be such that $w(i) = k_i\, m + i$. Then $(k_1,\ldots,k_{m-1})$ is a non-negative solution of the system

$$
\begin{aligned}
x_i &\geqslant 1 && \text{for all } i \in \{1,\ldots,m-1\},\\
x_i + x_j - x_{i+j} &\geqslant 0 && \text{for all } 1 \leqslant i \leqslant j \leqslant m-1,\ i+j \leqslant m-1,\\
x_i + x_j - x_{i+j-m} &\geqslant -1 && \text{for all } 1 \leqslant i \leqslant j \leqslant m-1,\ i+j > m\\
x_i &\in \mathbb{Z} && \text{for all } i \in \{1,\ldots,m-1\}
\end{aligned} \tag{19}
$$

Denote by $\mathcal{T}(m)$ the set of non-negative solutions of (19). Then, the following result, due to Rosales et al. [90], allow us to identify the set $\mathcal{T}(m)$ with $\mathcal{S}(m)$.

THEOREM 4.5.1. *Let m be an integer greater than 1. There exists a one-to-one correspondence map between the sets $\mathcal{T}(m)$ and $\mathcal{S}(m)$.*

Then, for fixed m, each solution of (19) corresponds with an unique numerical semigroup with multiplicity m.

The following result due to Selmer [94] allows us to define a polytope whose integer points represent each numerical semigroup with multiplicity and genus given.

THEOREM 4.5.2 ([94]). *Let S be a numerical semigroup with multiplicity m and $Ap(S,m) = \{0, w(1) = k_1 m + 1, \ldots, w(n-1) = k_{m-1} + m - 1\}$, then*

$$g(S) = \sum_{i=1}^{m-1} k_i$$

The system of inequalities

$$
\begin{aligned}
x_i &\geqslant 1 && \text{for all } i \in \{1,\ldots,m-1\},\\
x_i + x_j - x_{i+j} &\geqslant 0 && \text{for all } 1 \leqslant i \leqslant j \leqslant m-1,\ i+j \leqslant m-1,\\
x_i + x_j - x_{i+j-m} &\geqslant -1 && \text{for all } 1 \leqslant i \leqslant j \leqslant m-1,\ i+j > m\\
\sum_{i=1}^{m-1} x_i &= g
\end{aligned} \tag{20}
$$

defines a polytope in $\mathbb{R}^{m-1}$, that we will call $P_{m,g}$. We denote by $\mathcal{J}_{m,g}$ the system above.

Each element in $P_{m,g} \cap \mathbb{Z}$ corresponds with a numerical semigroup with multiplicity m and genus g.

We conclude that, for fixed m and g, the number of numerical semigroup with multiplicity m and genus g is finite. We will denote $n_{m,g} = \#(P_{m,g} \cap \mathbb{Z}^{m-1})$.

Once the problem of counting the number of semigroups for fixed multiplicity and genus, has been translated to count lattice points inside polytopes, Barvinok's rational function can help us.

Using Theorem 1.4.1 we can compute the following generating rational function for $P_{m,g}$

$$f(P_{m,g}; z) = \sum_{i \in I} \varepsilon_i \frac{z^{u_i}}{\prod_{j=1}^{n} (1 - z^{v_{ij}})} \tag{21}$$

in polynomial time.

Then, $n_{m,g} = \lim_{z \to 0} f(P_{m,g}; z)$.

This limit can be computed with the following with Algorithm 16.

We use the following result in Algorithm 16.

PROPOSITION 4.5.1. *Let $g(x) = \dfrac{p(x)}{q(x)}$, where $p(x)$ and $q(x)$ are polynomials in the indeterminate x, and such that 1 is a root of $q(x)$ with multiplicity r. If $\lim_{x \to 1} g(x)$ exists, then $p^{k)}(1) = 0$ for $k = 1, \ldots, r$.*

PROOF. Note that since 1 is a root of multiplicity r of $q(x)$, then we can write $q(x) = (1-x)^r h(x)$ with $h(x)$ a polynomial with $h(1) \neq 0$.

Assume that $p^{s)}(1) \neq 0$ for $s < r$. Computing the s-th derivative of q:

$$q^{r)}(x) = \frac{r!}{(r-k)!}(1-t)^{r-k} q(t) + \widetilde{q}(t)$$

with $\widetilde{q}(1) = 0$. Then, $\lim_{x \to 1} g(x) = \lim_{x \to 1} \dfrac{p(x)}{q(x)} = \cdots = \lim_{x \to 1} \dfrac{p^{k)}(x)}{q^{k)}(x)} = \dfrac{p^{k)}(1)}{0} = \infty$, a contradiction with the existence of limit of $g(x)$ at $x = 1$. □

Using the results on generating functions listed in Chapter 1 and the above proposition, the following result can be stated.

THEOREM 4.5.3. *For fixed multiplicity m, Algorithm 16 counts the numerical semigroups for any genus g in polynomial time.*

PROOF. By Theorem 1.4.1, Step 1 can be done in polynomial time. The rest of the steps are clearly polynomially bounded. □

Algorithm 16: Procedure for counting the number of numerical semigroups for fixed multiplicity and genus.

Input : $m, g \in \mathbb{Z}_+$, $1 \leq m \leq g+1$

Algorithm:

Step 1: Choose $c \in \mathbb{Z}^m$ such that $c\, v_{ij} \neq 0$ for all i, j. Do the changes $x_i = t^{c_i}$ for each $i = 1, \ldots, m$. Then equation (21) is expressed as:

$$f(P_{m,g}; t) = \sum_{i \in I} \varepsilon_i \frac{t^{c\, u_i}}{\prod_{j=1}^{n} (1 - t^{c\, v_{ij}})} \tag{22}$$

Step 2: Join the rational functions in equation (22) in one rational function:

$$f(P_{m,g}; t) = \frac{P(t)}{Q(t)} \tag{23}$$

Step 3: Extract from $Q(t)$ the terms with $(1-t)$, i.e., express $Q(t)$ as:

$$Q(t) = (1-t)^r\, q(t)$$

where $q(t)$ is a polynomial with $q(1) \neq 0$.

Step 4: Use L'Hopital Rule successively to compute the limit:

$$\lim_{z \to 1} f(P_{m,g}; z) = \lim_{t \to 1} f(P_{m,g}; t) = \lim_{t \to 1} \frac{P(t)}{(1-t)^r\, Q(t)} = \frac{P^{r)}(1)}{r! Q(1)}$$

Output: Number of numerical semigroups for fixed multiplicity m and genus g: $n_{m,g}$.

Another useful result is the one that states that for a numerical semigroup with fixed genus g, its multiplicity is at most $m = g+1$. Then, the number of semigroups with fixed genus g (independently of the multiplicity) is given by the following finite sum:

$$n_g = \sum_{m=1}^{g+1} n_{m,g}$$

THEOREM 4.5.4. *Let g be a given genus. $g+1$ iterations of Algorithm 16 counts the numerical semigroups of genus g in polynomial time.*

PROOF. It follows from Theorem 4.5.3. □

Table 4.5 shows the number of numerical semigroups from genus 0 to 15 obtained using the above methodology. We run the software `barvinok` [109] for counting $n_{m,g}$, for each pair (m, g) with $1 \leq m \leq g+1$ in a PC with an Intel Pentium 4 processor at 2.66GHz and 1 GB of RAM. It was able to compute the number of numerical semigroups for genus up to 15. Although this methodology can be applied to any genus, the software `barvinok` fails to compute the short generating function representation of the polytope in (20) for $g < 15$. `barvinok` is designed to be applicable to general polytopes and

therefore does not exploit the special structure of the polytope (20). Up to date, there are only few implementations for computing generating functions of rational polyhedrons. LattE [36] and `barvinok` [109] seems to be the most recent and incorporating a larger battery of options. More effective implementations oriented to the particular polytope in (20) and further research in the computation of generating functions of polyhedron will help to obtain the number of numerical semigroups for larger genuses. In Rosales et al. [90],

g \ m	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	n_g
2	1	1														2
3	1	2	1													4
4	1	2	3	1												7
5	1	2	4	4	1											12
6	1	3	6	7	5	1										33
7	1	3	7	10	11	6	1									39
8	1	3	9	13	17	16	7	1								67
9	1	4	11	16	27	28	22	8	1							118
10	1	4	13	22	37	44	44	29	9	1						204
11	1	4	15	24	49	64	72	66	37	10	1					343
12	1	5	18	32	66	85	116	116	95	46	11	1				592
13	1	5	20	35	85	112	172	188	182	132	56	12	1			1001
14	1	5	23	43	106	148	239	288	304	277	178	67	13	1		1693
15	1	6	26	51	133	191	325	409	492	486	409	234	79	14	1	2857

TABLE 4.2. Number of numerical semigroups with given genus g and multiplicity m for $g \leq 15$ and $2 \leq m \leq g+1$.

the authors proved that the set of numerical semigroups with fixed multiplicity and genus can be seen as the set of integer points in a certain rational bounded polyhedron. On other hand, the result of Barvinok [7] states that generating functions are useful to count integer points in rational polyhedron. This paper combines both results to count numerical semigroups for fixed multiplicity and genus. Furthermore, the result by Selmer [94] allows us to count the number of numerical semigroups just fixing the genus since this number only depends of the number of numerical semigroups with fixed genus g and multiplicity at most $g+1$.

A similar methodology can be applied to other special families of numerical semigroups where it is known their relationship with counting points in polyhedra. For instance, in [90] the authors characterized in this way maximal embedding dimension semigroups (MED-semigroups), symmetric numerical semigroups and maximal embedding dimension symmetric semigroups (MEDSY-semigroups). For all these cases, generating functions can be applied to compute the number of elements in the corresponding family fixing the genus.

CHAPTER 5

Non linear multiobjective optimization

In this chapter, we introduce a new methodology for solving general MOPIP (see Chapter 1 for a description of the problem) based on the construction of reduced Gröbner bases of certain ideals related to the problem and on solving triangular systems of polynomial equations given by those bases.

We describe different approaches for solving MOPIP using Gröbner bases which are based on reducing the problem to different optimality conditions: the necessary Karush-Kuhn-Tucker, the Fritz-John and the multiobjective Fritz-John optimality conditions.

In the following sections we describe some algorithms for solving MOPIP using tools from algebraic geometry. In particular, in each of these methods, we transform our problem to a certain system of polynomial equations, and we use Gröbner bases to solve it.

5.1. Obtaining nondominated solutions solving systems of polynomial equations

In this section we present the first approach for solving multiobjective polynomial integer programs using Gröbner bases. For this method, we transform the program in a system of polynomial equations that encodes the set of feasible solutions and its objective values. Solving that system in the objective values, and then, selecting the minimal ones in the partial componentwise order, allows us to obtain the associate feasible vectors, thus, the nondominated solutions.

Through this section we solve $MOPBP_{\mathbf{f},\mathbf{g},\mathbf{h}}$, i.e.

$$(MOPBP_{\mathbf{f},\mathbf{g},\mathbf{h}}) \qquad \begin{array}{lll} \min & (f_1(x),\ldots,f_k(x)) & \\ s.t. & g_j(x) \leq 0 & j=1,\ldots,m \\ & h_r(x) \leq 0 & r=1,\ldots,s \\ & x \in \{0,1\}^n & \end{array}$$

Without loss of generality, we reduce the general problem to the problem without inequality constraints since we can transform inequality constraints

to equality constrains as follows:

$$g(x) \leq 0 \Longleftrightarrow g(x) + z^2 = 0, z \in \mathbb{R}. \tag{24}$$

where the quadratic term, z^2, assures the nonnegativity of the slack variable and then, less than or equal to type inequality. Initially, we suppose that all the variables are binary. In Remark 5.1.1 we describe how to modify the algorithm to incorporate the above slack variables.

This approach consists of transforming $MOPBP_{\mathbf{f},\mathbf{h}}$ to an equivalent problem such that the objective functions are part of the constraints. For this transformation, we add k new variables, $y_1, \ldots, y_k$ to the problem, encoding the objective values for all feasible solutions. The modified problem is:

$$\begin{array}{lrll} \min & (y_1, \ldots, y_k) & & \\ s.t. & h_r(x) & = 0 & r = 1, \ldots, s \\ & y_j - f_j(x) & = 0 & j = 1, \ldots, k \\ & x_i(x_i - 1) & = 0 & i = 1, \ldots, n \\ & y \in \mathbb{R}^k & x \in \mathbb{R}^n & \end{array} \tag{25}$$

where integrality constraints are codified as quadratic constraints so $MOPBP_{\mathbf{f},\mathbf{h}}$ is a polynomial continuous problem.

The algorithm consists of, first, obtaining the set of feasible solutions of Problem (25) in the y variables; then, selecting from that set those solutions that are minimal with respect to the componentwise order, obtaining the set of efficient solutions of $MOPBP_{\mathbf{f},\mathbf{h}}$. The feasible solutions in the x-variables associated to those efficient solutions correspond with the nondominated solutions of $MOPBP_{\mathbf{f},\mathbf{h}}$.

Then, first, we concentrate in describing a procedure for solving the system of polynomial equations that encodes the feasible region of Problem (25), i.e. the solutions of

$$\begin{array}{lll} h_r(x) & = 0 & \text{for all } r = 1, \ldots, s \\ y_j - f_j(x) & = 0 & \text{for all } j = 1, \ldots, k \\ x_i(x_i - 1) & = 0 & \text{for all } i = 1, \ldots, n. \end{array} \tag{26}$$

For analyzing the system (26) we use Gröbner bases as a tool for solving systems of polynomial equations. Further details can be found in the book by Sturmfels [99].

The set of solutions of (26) coincides with the affine variety of the following polynomial ideal in $\mathbb{R}[y_1, \ldots, y_k, x_1, \ldots, x_n]$:

$$I = \langle h_1(x), \ldots, h_m(x), y_1 - f_1(x), \ldots, y_k - f_k(x), x_1(x_1 - 1), \ldots, x_n(x_n - 1) \rangle.$$

Note that I is a zero dimensional ideal since the number of solutions of the equations that define I is finite. Let $V(I)$ denote the affine variety of I. If we restrict I to the family of variables x (resp. y) the variety $V(I \cap \mathbb{R}[x_1, \ldots, x_n])$ (resp. $V(I \cap \mathbb{R}[y_1, \ldots, y_k])$) encodes the set of feasible solutions (resp. the set of possible objective values) for that problem.

Applying the elimination property, the reduced Gröbner basis for I, $\mathcal{G}$, with respect to the lexicographical ordering with $y_k \prec \cdots \prec y_1 \prec x_n \prec \cdots \prec x_1$ gives us a method for solving system (26) sequentially, i.e., solving in one indeterminate at a time. Explicitly, the shape of $\mathcal{G}$ is:

1) $\mathcal{G}$ contains one polynomial in $\mathbb{R}[y_k]$: $p_k(y_k)$
2) $\mathcal{G}$ contains one or several polynomials in $\mathbb{R}[y_{k-1}, y_k]$: $p_{k-1}^1(y_{k-1}, y_k), \ldots, p_{k-1}^{m_{k-1}}(y_{k-1}, y_k)$.

$\vdots$

$k+1$) $\mathcal{G}$ contains one or several polynomials in $\mathbb{R}[x_n, y_1, \ldots, y_k]$: $q_n^1(x_n, \mathbf{y}), \ldots, q_n^{s_n}(x_n, \mathbf{y})$.

$\vdots$

$k+n$) $\mathcal{G}$ contains one or several polynomials in $\mathbb{R}[x_n, y_1, \ldots, y_k]$: $q_1^1(x_1, \ldots, x_n, \mathbf{y}), \ldots, q_n^{s_1}(x_1, \ldots, x_n, \mathbf{y})$.

Then, with this structure of $\mathcal{G}$, we can solve, in a first step, the system in the y variables using those polynomial in $\mathcal{G}$ that only involve this family of variables as follows: we first solve for y_k in $p_k(y_k) = 0$, obtaining the solutions: $y_k^1, y_k^2, \ldots$. Then, for fixed y_k^r, we find the common roots of $p_{k-1}^1, p_{k-1}^2, \ldots$ getting solutions $y_{k-1,r}^1, y_{k-1,r}^2, \ldots$ and so on, until we have obtained the roots for $p_1(y_1, \ldots, y_k)$. Note that at each step we only solve one-variable polynomial equations.

We denote by Ω the above set of solutions in vector form

$$\Omega = \{(\hat{y}_1, \ldots, \hat{y}_k) : p_k(\hat{y}_k) = 0, p_{k-1}^1(\hat{y}_{k-1}, \hat{y}_k) = 0, \ldots, p_{k-1}^{m_{k-1}}(\hat{y}_{k-1}, \hat{y}_k) = 0, \ldots$$
$$p_1^1(\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_k) = 0, \ldots, p_1^{m_1}(\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_k) = 0\}.$$

As we stated above, Ω is the set of all possible values of the objective functions at the feasible solutions of $MOPBP_{\mathbf{f},\mathbf{h}}$. We are looking for the nondominated solutions that are associated with the efficient solutions. From Ω, we can select the efficient solutions as those that are minimal with respect to the componentwise order in $\mathbb{R}^k$. So, we can extract from Ω the set of efficient solutions, Y_E:

$$Y_E = \{(y_1^*, \ldots, y_k^*) \in \Omega : \nexists (y_1', \ldots, y_k') \in \Omega \text{ with } y_j' \leq y_j^* \text{ for } j = 1, \ldots, k \text{ and}$$
$$(y_1', \ldots, y_k') \neq (y_1^*, \ldots, y_k^*)\}$$

Once we have obtained the solutions in the y variables that are efficient solutions for $MOPBP_{\mathbf{f},\mathbf{h}}$, we compute with an analogous procedure the nondominated solutions associated to the y-values in Y_E. It consists of solving the triangular system given by $\mathcal{G}$ for the polynomial where the x-variables appear once the values for the y-variables are fixed to be each of the vectors in Y_E.

A pseudocode for this procedure is described in Algorithm 17.

Algorithm 17: Solving MOPIP by solving systems of polynomial equations

---

Input : $f_1, \ldots, f_k$, $h_1, \ldots h_s \in \mathbb{R}[x_1, \ldots, x_n]$

Initialization:

$I = \langle f_1 - y_1, \ldots, f_k - y_k, h_1, \ldots, h_s, x_1(x_1 - 1), \ldots, x_n(x_n - 1)\rangle$.

Algorithm:

Step 1.: Compute a Gröbner basis, G, for I with respect to a lexicographic order with $y_k \prec \cdots \prec y_1 \prec x_n \prec \cdots \prec x_1$.

Step 2.: Let $G_l^y = G \cap \mathbb{R}[y_{l+1}, \ldots, y_k]$ be a Gröbner basis for $I_l^y = I \cap \mathbb{R}[y_{l+1}, \ldots, y_k]$, for $l = 0, \ldots, k-1$.
(By the Elimination Property).

(1) Find all $\hat{y}_k \in V(G_{k-1}^y)$.

(2) Extend every $\hat{y}_k$ to $(\hat{y}_{k-1}, \hat{y}_k) \in V(G_{k-2}^y)$.

$\vdots$

$(k-1)$ Extend every $(\hat{y}_3, \ldots, \hat{y}_k)$ to $(\hat{y}_2, \hat{y}_3, \ldots, \hat{y}_k) \in V(G_1^y)$.

(k) Find all $\hat{y}_1$ such that $(\hat{y}_1, \ldots, \hat{y}_k) \in V(G_0^y)$.

Step 3.: Select from $V(G_0^y)$ the minimal vectors with respect to the usual componentwise order in $\mathbb{R}^k$. Set Y_E this subset.

Step 4.: Let $G_l = G \cap \mathbb{R}[y_1, \ldots, y_k, x_{l+1}, \ldots, x_n]$ a Gröbner basis for $I_l \cap \mathbb{R}[y_1, \ldots, y_k, x_{l+1}, \ldots, x_n]$, for $l = 0, \ldots, n-1$. (By the Elimination Property). Denote by
$S_l = \{(\hat{y}_1, \ldots, \hat{y}_k, \hat{x}_{l+1}, \ldots, \hat{x}_n) : (\hat{y}_1, \ldots, \hat{y}_k) \in$
Y_E, and $\exists (x_1, \ldots, x_l)$ such that $(x_1, \ldots, x_n)$ is feasible$\}$ for $l = 0, \ldots, n-1$.

(1) Find all $\hat{x}_n$ such that $(\hat{y}_1, \ldots, \hat{y}_k, \hat{x}_n) \in V(G_{n-1}) \cap S_{n-1}$.

(2) Extend every $\hat{x}_n$ to $((\hat{y}_1, \ldots, \hat{y}_k, \hat{x}_{n-1}, \hat{x}_n) \in V(G_{n-2}) \cap S_{n-2}$.

$\vdots$

$(n-1)$ Extend every $(\hat{y}_1, \ldots, \hat{y}_k, \hat{x}_3, \ldots, \hat{x}_n)$ to $(\hat{y}_1, \ldots, \hat{y}_k, \hat{x}_2, \hat{x}_3, \ldots, x_n) \in V(G_1) \cap S_1$.

(n) Find all $\hat{x}_1$ such that $(\hat{y}_1, \ldots, \hat{y}_k, \hat{x}_1, \ldots, \hat{x}_n) \in V(G_0) \cap S_0$.

Set $X_E = \pi_x(V(G_0) \cap S_0)$, where π_x denotes the projection over the x-variables.

Output: Y_E the set of efficient solutions and X_E the set of nondominated solutions for $MOPBP_{\mathbf{f},\mathbf{h}}$.

THEOREM 5.1.1. *Algorithm 17 either provides all nondominated and efficient solutions or provides a certificate of infeasibility whenever $G = \{1\}$.*

PROOF. Suppose that $G \neq \{1\}$. Then, G^y_{k-1} has exactly one element, namely $p(y_k)$. This follows from the observation that $I \cap \mathbb{R}[y_k]$ is a polynomial ideal in one variable, and therefore, needs only one generator.

Solving $p(y_k) = 0$ we obtain every $\hat{y}_k \in V(G^y_{k-1})$. Sequentially we obtain $\hat{y}_{k-1}$ extending $\hat{y}_k$ to the partial solutions $(\hat{y}_{k-1}, \hat{y}_k)$ in $V(G^y_{k-1})$ and so on.

By the Extension Theorem, this is always possible in our case.

Continuing in this way and applying the Extension Theorem, we can obtain all solutions $(\hat{y}_1, \ldots, \hat{y}_k)$ in $V(G \cap \mathbb{R}[y_1, \ldots, y_k]$. These vectors are all the possible objective values for all feasible solutions of the problem. Selecting from $V(G \cap \mathbb{R}[y_1, \ldots, y_k])$ those solutions that are not dominated in the componentwise order in $\mathbb{R}^k$, we obtain Y_E.

Following a similar scheme in the x- variables, we have the set $V(G_0) \cap S_0^*$ encoding all efficient (in the first k coordinates) and nondominated (in the last n coordinates) solutions.

Finally, if $G = \{1\}$, then, the ideal I coincides with $\mathbb{R}[y_1, \ldots, y_k, x_1, \ldots, x_n]$, indicating that $V(I)$ is empty (it is the set of the common roots of all polynomials in $\mathbb{R}[y_1, \ldots, y_k, x_1, \ldots, x_n]$). Then, we have an infeasible integer problem. □

REMARK 5.1.1. *In the case when we have added slack variables, as explained in (24), we slightly modify the above algorithm solving first in the slack variables and selecting those solutions that are real numbers. Then continue with the procedure described in Algorithm 17.*

The following example illustrates how Algorithm 17 works with the corresponding modifications due to non-binary variables and inequality constrains. This simple example gives an idea about the kind of problems that can be solved using this method. The feasible region has two different connected components and one of these components is not convex.

EXAMPLE 5.1.1.

$$
\begin{array}{lrl}
\min & (x_1^2 - x_2, x_1 - x_2^2) & \\
s.t. & x_2 - x_1^4 + 10x_1^3 - 30x_1^2 + 25x_1 - 7 & \geq 0 \\
& x_2 - x_1^3 + 9x_1^2 - 25x_1 + 12 & \leq 0 \\
& x_1, x_2 & \in \mathbb{Z}_+
\end{array} \tag{27}
$$

The feasible region for this problem is shown in Figure 5.1.

Now, we transform the problem to a binary problem and with equality constraints. For the first task, we substitute each variable, x_1 and x_2 by their binaries expressions, taking into account that we x_1 is upper bounded by 6 and x_2 by 15. For the second task, we add to the problem to new variables w_1 and w_2. Then, the problem is equivalent to the following multiobjective

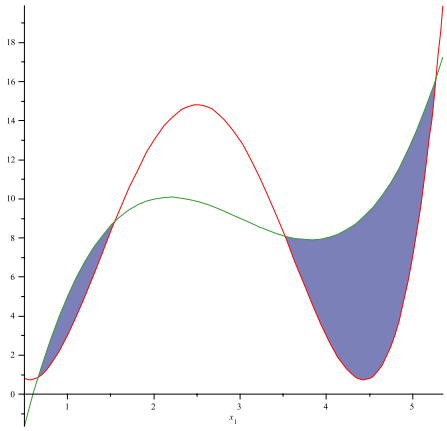


FIGURE 5.1. Feasible region of Example 5.1.1

mixed-binary problem:

$$\begin{aligned}
&\min && (y_1, y_2)\\
&s.t.\\
&\qquad y_1 - z_{10}^2 - 4\,z_{10}\,z_{11} - 8\,z_{10}\,z_{12} - 16\,z_{10}\,z_{13} - 4\,z_{11}^2\\
&\qquad -16\,z_{11}\,z_{12} - 32\,z_{11}\,z_{13} - 16\,z_{12}^2 - 64\,z_{12}\,z_{13} - 64\,z_{13}^2\\
&\qquad +z_{20} + 2\,z_{21} + 4\,z_{22} + 8\,z_{23} + 16\,z_{24} \quad = 0\\
&\qquad y_2 - z_{10} - 2\,z_{11} - 4\,z_{12} - 8\,z_{13} + z_{20}^2 + 4\,z_{20}\,z_{21} + 8\,z_{20}\,z_{22}\\
&\qquad +16\,z_{20}\,z_{23} + 32\,z_{20}\,z_{24} + 4\,z_{21}^2 + 16\,z_{21}\,z_{22} + 32\,z_{21}\,z_{23}\\
&\qquad +64\,z_{21}\,z_{24} + 16\,z_{22}^2 + 64\,z_{22}\,z_{23} + 128\,z_{22}\,z_{24} + 64\,z_{23}^2\\
&\qquad +256\,z_{23}\,z_{24} + 256\,z_{24}^2 \quad = 0
\end{aligned}$$

$$\begin{aligned}
&-7 - 1536\,z_{10}\,z_{11}\,z_{13}^2 - 192\,z_{10}^2\,z_{11}\,z_{13} - 1536\,z_{10}\,z_{12}^2\,z_{13}\\
&-192\,z_{10}\,z_{11}^2\,z_{12} + 25\,z_{10} + 50\,z_{11} + 100\,z_{12} + 200\,z_{13} - 30\,z_{10}^2\\
&-120\,z_{10}\,z_{11} - 240\,z_{10}\,z_{12} - 480\,z_{10}\,z_{13} - 120\,z_{11}^2 - 480\,z_{11}\,z_{12} -\\
&960\,z_{11}\,z_{13} - 480\,z_{12}^2 - 1920\,z_{12}\,z_{13} - 1920\,z_{13}^2 + z_{20} + 2\,z_{21} + 4\,z_{22}\\
&+8\,z_{23} + 16\,z_{24} + 960\,z_{10}\,z_{11}\,z_{13} + 1920\,z_{10}\,z_{12}\,z_{13} + 60\,z_{10}^2\,z_{11}\\
&+120\,z_{10}^2\,z_{12} + 240\,z_{10}^2\,z_{13} + 120\,z_{10}\,z_{11}^2 + 480\,z_{10}\,z_{12}^2 + 1920\,z_{10}\,z_{13}^2\\
&+480\,z_{11}^2\,z_{12} + 960\,z_{11}^2\,z_{13} + 960\,z_{11}\,z_{12}^2 + 3840\,z_{11}\,z_{13}^2 + 10\,z_{10}^3\\
&+80\,z_{11}^3 + 640\,z_{12}^3 + 5120\,z_{13}^3 - 1536\,z_{10}\,z_{11}\,z_{12}\,z_{13} - z_{10}^4 - 256\,z_{12}^4\\
&-4096\,z_{13}^4 - 16\,z_{11}^4 + 3840\,z_{12}^2\,z_{13} + 7680\,z_{12}\,z_{13}^2 - 8\,z_{10}^3\,z_{11}\\
&-16\,z_{10}^3\,z_{12} - 32\,z_{10}^3\,z_{13} - 24\,z_{10}^2\,z_{11}^2 - 96\,z_{10}^2\,z_{12}^2\\
&-384\,z_{10}^2\,z_{13}^2 - 32\,z_{10}\,z_{11}^3 - 256\,z_{10}\,z_{12}^3 - 2048\,z_{10}\,z_{13}^3\\
&-128\,z_{11}^3\,z_{12} - 256\,z_{11}^3\,z_{13} - 384\,z_{11}^2\,z_{12}^2 - 1536\,z_{11}^2\,z_{13}^2\\
&-512\,z_{11}\,z_{12}^3 - 4096\,z_{11}\,z_{13}^3 - 8192\,z_{12}\,z_{13}^3 - 2048\,z_{12}^3\,z_{13}
\end{aligned}$$

$$\begin{aligned}
-6144\,z_{12}^2\,z_{13}^2 + 3840\,z_{11}\,z_{12}\,z_{13} - 384\,z_{10}^2\,z_{12}\,z_{13} - 96\,z_{10}^2\,z_{11}\,z_{12} \\
-1536\,z_{11}^2\,z_{12}\,z_{13} - 384\,z_{10}\,z_{11}^2\,z_{13} - 384\,z_{10}\,z_{11}\,z_{12}^2 \\
-3072\,z_{10}\,z_{12}\,z_{13}^2 - 3072\,z_{11}\,z_{12}^2\,z_{13} - 6144\,z_{11}\,z_{12}\,z_{13}^2 \\
+480\,z_{10}\,z_{11}\,z_{12} - w_1^2 \quad = 0
\end{aligned}$$

$$\begin{aligned}
12 - 25\,z_{10} - 50\,z_{11} - 100\,z_{12} - 200\,z_{13} + 9\,z_{10}^2 + 36\,z_{10}\,z_{11} \\
+72\,z_{10}\,z_{12} + 144\,z_{10}\,z_{13} + 36\,z_{11}^2 + 144\,z_{11}\,z_{12} \\
+288\,z_{11}\,z_{13} + 144\,z_{12}^2 + 576\,z_{12}\,z_{13} + 576\,z_{13}^2 + z_{20} + 2\,z_{21} \\
+4\,z_{22} + 8\,z_{23} + 16\,z_{24} - 96\,z_{10}\,z_{11}\,z_{13} - 192\,z_{10}\,z_{12}\,z_{13} \\
-6\,z_{10}^2\,z_{11} - 12\,z_{10}^2\,z_{12} - 24\,z_{10}^2\,z_{13} \\
-12\,z_{10}\,z_{11}^2 - 48\,z_{10}\,z_{12}^2 - 192\,z_{10}\,z_{13}^2 - 48\,z_{11}^2\,z_{12} \\
-96\,z_{11}^2\,z_{13} - 96\,z_{11}\,z_{12}^2 - 384\,z_{11}\,z_{13}^2 - z_{10}^3 - 8\,z_{11}^3 \\
-64\,z_{12}^3 - 512\,z_{13}^3 - 384\,z_{12}^2\,z_{13} - 768\,z_{12}\,z_{13}^2 \\
-384\,z_{11}\,z_{12}\,z_{13} - 48\,z_{10}\,z_{11}\,z_{12} + w_2^2 \quad = 0 \\
z_{ij}^2 - z_{ij} = 0, \quad i = 1, 2 j = 0, 1, 2, 3, (4) \\
zij, y_k, w_i \in \mathbb{R}.
\end{aligned}$$

Now, we compute the reduced Gröbner basis for the set of polynomials that define the feasible region of the above problem with respect to the lexicographic ordering such that $\mathbf{z} \succ \mathbf{yw}$. *Running MAPLE 11 using the package* `Groebner` *and the procedure* `Solve`, *it computes the* 569 *complex solutions solutions of the polynomial system. Then, we remove those that are not real constants (it may happen only for the slack variables* w_1 *and* w_2*). After this removing process we have* 29 *solutions (16 of them given different values in the y-variables, note that oposite values for the w-variables give the same y-values) in the* 13 *variables that are involved in the problem. Actually, we can forget the values for variables* w_1 *and* w_2 *since the were only useful to determine the real solutions. After that, we have the following solutions for the y variables:*

$$\begin{aligned}
(y_1, y_2) \in \{&(16, -76), (9, -45), (17, -59), (14, -116), (11, -21), (13, -139), \\
&(-3, -15), (13, -5), (-2, -8), \ (12, -12), (12, -164), (15, -95), \\
&(10, -32), (18, -44), (8, -60), \ (-4, -24)\}.
\end{aligned}$$

The minimal elements (with respect to the componentwise ordering) are:

$$Y_E = \{(12, -164), (8, -60), (-4, -24)\},$$

whose values in the z variables are:

$$Z_E = \{(0, 0, 1, 0, 0, 0, 0, 1, 0), (1, 0, 0, 0, 1, 0, 1, 0, 0), (1, 0, 1, 0, 1, 0, 1, 1, 0)\},$$

and translating to values in the original x variables we have that the set of nondominated solutions for the problem is:

$$X_E = \{(4, 8), (5, 13), (1, 5)\}$$

Figure 5.1.1 shows these solutions in the feasible region of the problem and the levels curves of both objective functions at each of these solutions.

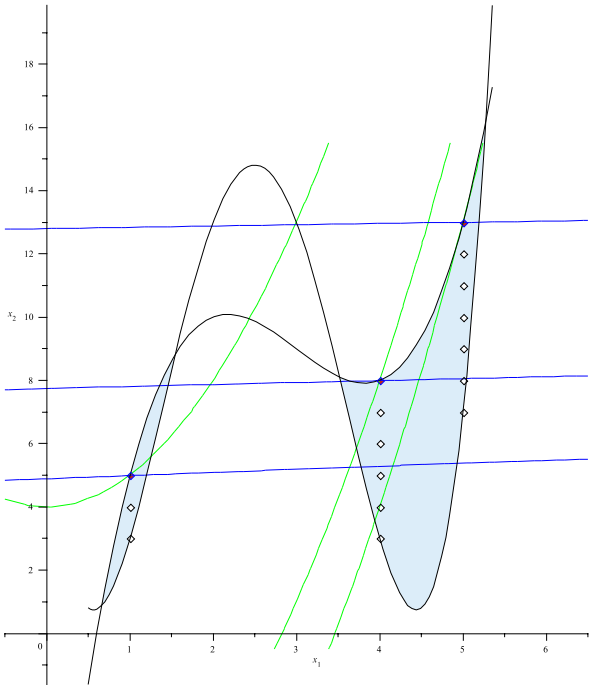


FIGURE 5.2. Feasible region, the nondominated solutions and the level curves of Example 5.1.1

REMARK 5.1.2. *The Gröbner basis, $\mathcal{G}$, computed for solving the system of polynomial equations can be computed with respect to any other elimination ordering. The only conditions required for that ordering is that it allows to separate the family of $x-$variables from the family of y-variables and such that the system of polynomials given by that basis allows solving first for the y-variables and then for the x-variables sequentially.*

5.2. Obtaining nondominated solutions by the Chebyshev norm approach

In this section we describe two more methods for solving MOPIP based on a different rationale, namely scalarizing the multiobjective problem and solving it as a parametric single-objective problem. We propose a methodology based on the application of optimality conditions to a family of single-objective problems related to our original multiobjective problem. The methods consist

of two main steps: a first step where the multiobjective problem is scalarized to a family of single-objective problems such that each nondominated solution is an optimal solution for at least one of the single-objective problems in that family; and a second step that consists of applying necessary optimality conditions to each one of the problems in the family, to obtain their optimal solutions. Those solutions are only candidates to be nondominated solutions of the multiobjective problem since we just use necessary conditions.

For the first step, the scalarization, we use a weighted Chebyshev norm approach. Other weighted sum approaches could be used to transform the multiobjective problem in a family of single-objective problems whose set of solutions contains the set of nondominated solutions of our problem. However, the Chebyshev approach seems to be rather adequate since it does not require to impose extra hypothesis to the problem. This approach can be improved for problems satisfying convexity conditions, where alternative well-known results can be applied (see [63] for further details).

For the second step, we use the Fritz-John and Karush-Kuhn-Tucker necessary optimality conditions, giving us two different approaches. In this section we describe both methodologies since each of them has its own advantages over the other.

For applying the Chebyshev norm scalarization, we use the following result that states how to transform our problem to a family of single-objective problems, and how to obtain nondominated solutions from the optimal solution of those single-objective problems. Further details and proofs of this result can be found in [63].

THEOREM 5.2.1 (Corollary 11.21 in [63]). *Let $MOPBP_{\mathbf{f},\mathbf{g},\mathbf{h}}$ be feasible. x^* is a nondominated solution of $MOPBP_{\mathbf{f},\mathbf{g},\mathbf{h}}$ if and only if there are positive real numbers $\omega_1, \ldots, \omega_k > 0$ such that x^* is an image unique solution of the following weighted Chebyshev approximation problem:*

$$
(P_\omega) \qquad \begin{array}{lrll} \min & \gamma & & \\ s.t. & \omega_i \left(f_i(x) - \hat{y}_i\right) - \gamma & \leq 0 & i = 1, \ldots, k \\ & g_j(x) & \leq 0 & j = 1, \ldots, m \\ & h_r(x) & = 0 & r = 1, \ldots, s \\ & x_i(x_i - 1) & = 0 & i = 1, \ldots, n \\ & \gamma \in \mathbb{R} & x \in \mathbb{R}^n & \end{array}
$$

where $\hat{y} = (\hat{y}_1, \ldots, \hat{y}_k) \in \mathbb{R}^k$ is a lower bound of $f = (f_1, \ldots, f_k)$, i.e., $\hat{y}_i \leq f_i(x)$ for all feasible solution x and $i = 1, \ldots, k$.

According to the above result, every nondominated solution of $MOPBP_{\mathbf{f},\mathbf{g},\mathbf{h}}$ is the unique solution of P_ω for some $\omega > 0$. We apply, in the second step,

necessary optimality conditions for obtaining the optimal solutions for those problems (taking ω as parameters). These solutions are candidates to be non-dominated solutions of our original problem. Actually, every nondominated solution is among those candidates.

In the following subsections we describe the above-mentioned two methodologies for obtaining the optimal solutions for the scalarized problems P_ω for each ω.

5.2.1. The Chebyshev-Karush-Kuhn-Tucker approach: The first optimality conditions that we apply are the Karush-Kuhn-Tucker (KKT) necessary optimality conditions, that were stated, for the general case, as follows (see e.g. [10] for further details):

THEOREM 5.2.2 (KKT necessary conditions). *Consider the problem:*

$$\begin{array}{lrll} \min & f(x) & & \\ s.t. & g_j(x) & \leq 0 & = 1,\dots,m \\ & h_r(x) & = 0 & r = 1,\dots,s \\ & x & \in \mathbb{R}^n & \end{array} \tag{28}$$

Let x^* be a feasible solution, and let $J = \{j : g_j(x^*) = 0\}$. Suppose that f and g_j, for $j = 1,\dots,m$, are differentiable at x^*, that g_j, for $j \notin J$, is continuous at x^*,and that h_r, for $r = 1,\dots,s$, is continuously differentiable at x^*. Further suppose that ∇g_j, for $j \in I$, and ∇h_r, for $r = 1,\dots,s$, are linearly independent (regularity conditions). If x^* solves Problem 28 locally, then there exist scalars λ_j, for $j = 1,\dots,m$, and μ_r, for $r = 1,\dots,s$, such that

$$\begin{array}{rcll} \nabla f(x^*) + \sum_{j=1}^{m} \lambda_j \nabla g_j(x^*) + & \sum_{r=1}^{s} \mu_r \nabla h_r(x^*) & = 0 & \\ \lambda_j \, g_j(x^*) & = & 0 & \text{for } j = 1,\dots,m \\ \lambda_j & \geq & 0 & \text{for } j = 1,\dots,m \end{array} \tag{KKT}$$

From the above theorem the candidates to be optimal solutions for Problem (28) are those that either satisfy the KKT conditions (in the case where all the functions involved in Problem (28) are polynomials, this is a system of polynomial equations) or do not satisfy the regularity conditions. Note that these two sets are, in general, not disjoint.

Regularity conditions can also be formulated as a system of polynomial equations when the involved functions are all polynomials. Let x^* be a feasible solution for Problem (28), x^* does not verify the regularity conditions if there exist scalars λ_j, for $j \in J$, and μ_r, for $r = 1,\dots,s$, not all equal to zero, such

that:

$$\text{(Non-Regularity)} \qquad \sum_{j\in I} \lambda_j \nabla g_j + \sum_{r=1}^{s} \mu_r \nabla h_r = 0$$

The above discussion justifies the following result.

COROLLARY 5.2.1. *Let x^* be a nondominated solution for $MOPBP_{\mathbf{f},\mathbf{g},\mathbf{h}}$. Then, x^* is a solution of the systems of polynomial equations (29) or (30), for some $\omega > 0$.*

$$(29)\left.\begin{array}{r} 1 - \sum_{i=1}^{k} \nu_i = 0 \\ \sum_{i=1}^{k} \nu_i \omega_i \nabla f_i(x) + \sum_{j=1}^{m} \lambda_j \nabla g_j(x) + \sum_{r=1}^{s} \mu_j \nabla h_r(x) + \sum_{l=1}^{n} \beta_l \, \delta_{il}(2x_i - 1) = 0 \\ \nu_i \, \omega_i \, (f_i(x) - \hat{y}_i) - \gamma = 0 \\ i = 1, \ldots, k \end{array}\right\}$$

for some $\lambda_j \, g_j(x) = 0$, for $j = 1, \ldots, m$, $\lambda_j \geq 0$, for $j = 1, \ldots, m$, and $\nu_i \geq 0$, for $i = 1, \ldots, k$.

$$(30)\left.\begin{array}{r} \sum_{i=1}^{k} \nu_i = 0 \\ \sum_{i=1}^{k} \nu_i \omega_i \nabla f_i(x) + \sum_{j=1}^{m} \lambda_j \nabla g_j(x) + \sum_{r=1}^{s} \mu_j \nabla h_r(x) + \sum_{l=1}^{n} \beta_l \, \delta_{il}(2x_i - 1) = 0 \\ \omega_i \, (f_i(x) - \hat{y}_i) - \gamma \leq 0 \\ i = 1, \ldots, k \end{array}\right\}$$

with $x \in \mathbb{R}^n$ such that $g_j(x) \leq 0$, for $j = 1, \ldots, m$, $h_r(x) = 0$, for $r = 1, \ldots, s$ $\lambda_j \geq 0$, for $j = 1, \ldots, m$, and $\nu_i \geq 0$, for $i = 1, \ldots, k$.

Let X_E^{KKT} denote the set of solutions, in the x-variables, of system (29) and let X_E^{NR} denote the set of solutions, in the x-variables, of system (30) (the problem is solved avoiding inequality constraints, then every solution is evaluated to check if it satisfies the inequality constraints).

For solving these systems (Chebyshev-KKT and Non-Regulariry), we use a Gröbner bases approach. Let I be the ideal generated by the involved equations.

Let us consider a lexicographical order over the monomials in $\mathbb{R}[\mathbf{x}, \gamma, \lambda, \nu, \mu, \beta]$ such that $\mathbf{x} \prec \gamma \prec \lambda \prec \nu \prec \mu \prec \beta$. Then, the Gröbner basis, $\mathcal{G}$, for I with this order has the following triangular shape:

- $\mathcal{G}$ contains one polynomial in $\mathbb{R}[x_n]$: $p_n(x_n)$
- $\mathcal{G}$ contains one or several polynomials in $\mathbb{R}[x_{n-1}, x_n]$: $p_{n-1}^1(x_{n-1}, x_n), \ldots, p_{n-1}^{m_1}(x_{n-1}, x_n)$.

$\vdots$

- $\mathcal{G}$ contains one or several polynomials in $\mathbb{R}[\mathbf{x}]$: $p_1^1(x_1,\ldots,x_n),\ldots,p_1^{m_n}(x_1,\ldots,x_n)$.
- The remaining polynomials involve variables $\mathbf{x}$ and at least one γ, λ, μ, ν or β.

We are interested in finding only the values for the x-variables, so, we avoid the polynomials in $\mathcal{G}$ that involve any of the other auxiliary variables. In general, we are not able to discuss about the values of the parameters γ, λ, μ, ν and β. Needless to say that in those cases when we can do it, some values of $\mathbf{x}$ may be discarded simplifying the process. We denote by $\mathcal{G}^x$ the subset of $\mathcal{G}$ that contains only polynomials in the x-variables. By the Extension Theorem, $\mathcal{G}^x$ is a Gröbner basis for $I\cap\mathbb{R}[x_1,\ldots,x_n]$.

Solving the system given by $\mathcal{G}^x$ and checking feasibility of those solutions, we obtain as solutions those of our KKT or Non-Regularity original systems.

It is clear that the set of nondominated solutions of our problem is a subset of $X_E^{KKT}\cup X_E^{NR}$, since either a solution is regular, and then, KKT conditions are applicable or it satisfies the non regularity conditions. However, the set $X_E^{KKT}\cup X_E^{NR}$ may contain dominated solutions, so, at the end we must remove the dominated ones to get only X_E.

The steps to solve Problem $MOPBP_{\mathbf{f},\mathbf{g},\mathbf{h}}$ using the Chebyshev-KKT approach are summarized in Algorithm 18.

Algorithm 18: Summary of the procedure for solving MOPBP using Chebyshev-KKT approach.

Input : $f_1,\ldots,f_k,\ g_1,\ldots g_m,h_1,\ldots,h_r\in\mathbb{R}[x_1,\ldots,x_n]$

Algorithm:

Step 1: Formulate the Chebyshev scalarization of $MOPBP_{\mathbf{f},\mathbf{g},\mathbf{h}}$. (Problem P_ω)

Step 2: Solve System (29) in the x-variables: X_E^{KKT}.

Step 3: Solve System (30) in the x-variables: X_E^{NR}.

Step 4: Remove from $X_E^{KKT}\cup X_E^{NR}$ the subset of dominated solutions: X_E.

Output: X_E the set of nondominated solutions for $MOPBP_{\mathbf{f},\mathbf{g},\mathbf{h}}$

THEOREM 5.2.3. *Algorithm 18 solves Problem $MOPBP_{\mathbf{f},\mathbf{g},\mathbf{h}}$ in a finite number of steps.*

The following example illustrates the how the algorithm works.

EXAMPLE 5.2.1. *Consider the following simple problem in two variables:*

$$\begin{array}{lll} \min & (8x_1 - 8x_2 + 2x_1x_2, -2x_1 + 5x_2 + 4x_1x_2) & \\ s.t. & -x_1 + x_1^3 + x_2^4 \geq 0 & \\ & x_1, x_2 & \in \{0,1\} \end{array} \tag{31}$$

The Chebyshev scalarization of this problem is:

$$\begin{array}{ll} \min & \gamma \\ s.t. & \omega_1 \left(8x_1 - 8x_2 + 2x_1x_2 - (-2)\right) - \gamma \leq 0 \\ & \omega_2 \left(-2x_1 + 5x_2 + 4x_1x_2 - (-2)\right) - \gamma \leq 0 \\ & x_1 - x_1^3 - x_2^4 \leq 0 \\ & \gamma, \omega_1, \omega_2 \in \mathbb{R}, x_1, x_2 \in \{0,1\} \end{array} \tag{32}$$

Then, KKT system is the one given by the partial derivatives of the Lagrangean operator:

(33)

$$\begin{aligned} \mu_1 \left(\omega_1 \left(8\,x_1 - 8\,x_2 + 2\,x_1\,x_2 + 8\right) - \gamma\right) &= 0 \\ \mu_2 \left(\omega_2 \left(-2\,x_1 + 5\,x_2 + 4\,x_1\,x_2 + 2\right) - \gamma\right) &= 0 \\ \lambda_1 \left(2\,x_1 - 1\right) + \mu_1\,\omega_1 \left(8 + 2\,x_2\right) + \mu_2\,\omega_2 \left(-2 + 4\,x_2\right) - \nu \left(-1 + 3\,x_1^2\right) &= 0 \\ \lambda_2 \left(2\,x_2 - 1\right) + \mu_1\,\omega_1 \left(-8 + 2\,x_1\right) + \mu_2\,\omega_2 \left(5 + 4\,x_1\right) - 4\,\nu\,x_2^3 &= 0 \\ 1 - \mu_1 - \mu_2 &= 0 \\ \nu \left(-x_1 + x_1^3 + x_2^4\right) &= 0 \\ x_1^2 - x_1 &= 0 \\ x_2^2 - x_2 &= 0 \\ \mu_1 &\geq 0 \\ \mu_2 &\geq 0 \\ \nu &\geq 0 \end{aligned}$$

We solve this system sequentially using a Gröbner basis with respect to a lexicographic ordering with $\mu \prec \nu \prec x \prec \gamma \prec \lambda$. With this order we obtain the following Gröbner basis:

$\mathcal{G} = \{-1+\mu_1+\mu_2, \nu\,x_2, x_2^2-x_2, x_1^2-x_1, \mu_2\,\gamma\,x_2-\mu_2^2\,\gamma\,x_2-\mu_2\,\gamma\,x_1+\mu_2^2\,\gamma\,x_1, \mu_2\,\gamma\,\nu\,x_1, \gamma\,x_1\,x_2-$
$x_2\,\gamma+\mu_2\,\gamma\,x_2-\mu_2\,\gamma\,x_1, 10\,\lambda_2\,x_2-10\,\gamma-10\,\lambda_2-6\,x_2\,\gamma-19\,\mu_2\,\gamma\,x_2+10\,x_1\,\lambda_2+$
$35\,\mu_2\,\gamma+10\,x_1\,\gamma, -273\,\gamma^2\,x_2+1173\,\mu_2\,\gamma^2\,x_2+280\,\gamma^2-980\,\mu_2\,\gamma^2-175\,\gamma^2\,x_1+$
$255\,\mu_2\,\gamma^2\,x_1+280\,\lambda_2\,\gamma, -259\,\lambda_2\,x_2-252\,\nu-308\,\gamma-56\,\lambda_2+252\,\lambda_1+336\,x_2\,\gamma-$
$1025\,\mu_2\,\gamma\,x_2-252\,\nu\,x_1+413\,x_1\,\gamma-375\,\mu_2\,\gamma\,x_1+700\,\mu_2\,\gamma, 56\,\lambda_2\,x_2-56\,\gamma-$
$56\,\lambda_2+21\,x_2\,\gamma+19\,\mu_2\,\gamma\,x_2+504\,\mu_2\,\omega_2+35\,x_1\,\gamma-19\,\mu_2\,\gamma\,x_1-56\,\mu_2\,\gamma, 560\,\omega_1\,\mu_2+$
$70\,\lambda_2\,x_2-560\,\omega_1-21\,x_2\,\gamma+71\,\mu_2\,\gamma\,x_2-35\,x_1\,\gamma+55\,\mu_2\,\gamma\,x_1+70\,\gamma-70\,\mu_2\,\gamma, -\gamma^2\,x_1\,\nu+$
$16\,\gamma\,\omega_1\,x_1\,\nu, 8960\,\gamma\,\omega_1\,x_1+511\,\gamma^2\,x_2-18223\,\mu_2\,\gamma^2\,x_2+3920\,\gamma^2+13720\,\mu_2\,\gamma^2+$
$22400\,\gamma\,\omega_1\,x_2-35280\,\omega_2\,\gamma-1295\,\gamma^2\,x_1-1769\,\mu_2\,\gamma^2\,x_1+10584\,\gamma\,\omega_2\,x_2-31360\,\omega_1\,\lambda_2+$
$282240\,\omega_2\,\omega_1-35280\,\lambda_2\,\omega_2\,x_2+31360\,\lambda_2\,\omega_1\,x_2+17640\,\gamma\,\omega_2\,x_1-62720\,\omega_1\,\gamma\}$

Sequentially solving, we obtain the following sets of solutions (they depend of some variables, since the sysem is not zero-dimensional): $\{\omega_2 = \omega_2, \lambda_1 = 2\,\omega_2, \lambda_2 = 9\,\omega_2, \gamma = 0, x_2 = 0, \mu_1 = 0, \nu = 0, \mu_2 = 1, x_1 = 1, \omega_1 = \omega_1\}, \{\omega_2 = \omega_2, \lambda_2 = -6\,\omega_1, \lambda_1 = -8\,\omega_1, \mu_1 = 1, \gamma = 16\,\omega_1, \mu_2 = 0, x_2 = 0, \nu = 0, x_1 = 1, \omega_1 = \omega_1\}, \{\omega_2 = \omega_2, \lambda_1 = -2\,\omega_2, \gamma = 2\,\omega_2, \lambda_2 = 5\,\omega_2, x_1 = 0, x_2 = 0, \mu_1 = 0, \nu = 0, \mu_2 = 1, \omega_1 = \omega_1\}, \{\omega_2 = \omega_2, \lambda_1 = -2\,\omega_2, x_2 = 1, \gamma = 9\,\omega_2, \lambda_2 = -9\,\omega_2, \mu_1 = 0, \nu = 0, \mu_2 = 1, x_1 = 1, \omega_1 = \omega_1\}, \{\nu = \nu, \lambda_2 = \lambda_2, \mu_2 = \mu_2, x_1 = 0, x_2 = 0, \mu_1 = 1 - 1\,\mu_2, \gamma = 2\,\lambda_2/(-2 + 7\,\mu_2), \lambda_1 = -(1\,(-2\,\lambda_2 + 2\,\nu + 4\,\mu_2\,\lambda_2 - 7\,\mu_2\,\nu))/(-2 + 7\,\mu_2), \omega_2 = \lambda_2/(-2 + 7\,\mu_2), \omega_1 = 2500000000\,\lambda_2/(-2 + 7\,\mu_2)\}, \{x_2 = 1, \lambda_2 = \lambda_2, \mu_2 = \mu_2, \nu = 0, x_1 = 1, \mu_1 = 1 - 1\,\mu_2, \gamma = -5\,\lambda_2/(-3 + 8\,\mu_2), \lambda_1 = -5555555556\,\lambda_2\,(-9 + 7\,\mu_2)/(-3 + 8\,\mu_2), \omega_2 = -5555555556\,\lambda_2/(-3 + 8\,\mu_2), \omega_1 = -5000000000\,\lambda_2/(-3 + 8\,\mu_2)\}, \{\omega_2 = \omega_2, \nu = \nu, \mu_2 = \mu_2, \gamma = 0, x_2 = 0, x_1 = 1, \mu_1 = 1 - 1\,\mu_2, \omega_1 = 0, \lambda_2 = 9\,\mu_2\,\omega_2, \lambda_1 = 2\,\mu_2\,\omega_2 + 2\,\nu\}, \{x_2 = 1, \nu = 0, x_1 = 1, \mu_1 = 6250000000, \omega_1 = -1411764706\,\lambda_1, \mu_2 = 3750000000, \gamma = -1411764706\,\lambda_1, \lambda_2 = 0, \lambda_1 = \lambda_1, \omega_2 = -1568627451\,\lambda_1\}, \{\omega_2 = \omega_2, \mu_1 = 1, x_1 = 0, \mu_2 = 0, x_2 = 0, \omega_1 = \omega_1, \lambda_1 = \lambda_1, \lambda_2 = -8\,\omega_1, \nu = -8\,\omega_1 + \lambda_1, \gamma = 8\,\omega_1\}, \{\omega_2 = \omega_2, x_2 = 1, \mu_1 = 1, x_1 = 0, \mu_2 = 0, \gamma = 0, \nu = 0, \omega_1 = \omega_1, \lambda_2 = 8\,\omega_1, \lambda_1 = 10\,\omega_1\}, \{x_2 = 1, \mu_2 = \mu_2, x_1 = 0, \gamma = 0, \nu = 0, \omega_1 = \omega_1, \mu_1 = 1 - 1\,\mu_2, \omega_2 = 0, \lambda_2 = -8\,\omega_1\,\mu_2 + 8\,\omega_1, \lambda_1 = -10\,\omega_1\,\mu_2 + 10\,\omega_1\}, \{\nu = \nu, x_1 = 0, x_2 = 0, \lambda_2 = 0, \lambda_1 = \lambda_1, \gamma = 2333333333\,\lambda_1 - 2333333333\,\nu, \omega_2 = 1166666667\,\lambda_1 - 1166666667\,\nu, \mu_1 = 7142857143, \mu_2 = 2857142857, \omega_1 = 2916666667\,\lambda_1 - 2916666667\,\nu\}, \{\lambda_2 = \lambda_2, \mu_2 = \mu_2, x_1 = 0, x_2 = 0, \nu = 0, \mu_1 = 1 - 1\,\mu_2, \gamma = 2\,\lambda_2/(-2 + 7\,\mu_2), \omega_2 = \lambda_2/(-2 + 7\,\mu_2), \omega_1 = 2500000000\,\lambda_2/(-2 + 7\,\mu_2), \lambda_1 = -2\,\lambda_2\,(-1 + 2\,\mu_2)/(-2 + 7\,\mu_2)\}, \{\omega_2 = \omega_2, \nu = \nu, \lambda_2 = -6\,\omega_1, \mu_1 = 1, \gamma = 16\,\omega_1, \mu_2 = 0, x_2 = 0, x_1 = 1, \omega_1 = \omega_1, \lambda_1 = -8\,\omega_1 + 2\,\nu\}, \{\omega_2 = \omega_2, \gamma = 2\,\omega_2, \nu = \nu, \lambda_2 = 5\,\omega_2, x_1 = 0, x_2 = 0, \mu_1 = 0, \mu_2 = 1, \omega_1 = \omega_1, \lambda_1 = -2\,\omega_2 + \nu\}, \{\omega_2 = \omega_2, x_2 = 1, \mu_1 = 1, \mu_2 = 0, \nu = 0, x_1 = 1, \omega_1 = \omega_1, \lambda_2 = 6\,\omega_1, \lambda_1 = -10\,\omega_1, \gamma = 10\,\omega_1\}, \{x_1 = 0, x_2 = 0, \nu = 0, \lambda_2 = 0, \lambda_1 = \lambda_1, \mu_1 = 7142857143, \mu_2 = 2857142857, \gamma = 2333333333\,\lambda_1, \omega_1 = 2916666667\,\lambda_1, \omega_2 = 1166666667\,\lambda_1\}, \{\omega_2 = \omega_2, \nu = \nu, \lambda_2 = 9\,\omega_2, \gamma = 0, x_2 = 0, \mu_1 = 0, \mu_2 = 1, x_1 = 1, \omega_1 = \omega_1, \lambda_1 = 2\,\omega_2 + 2\,\nu\}, \{\omega_2 = \omega_2, \mu_2 = \mu_2, \gamma = 0, x_2 = 0, \nu = 0, x_1 = 1, \mu_1 = 1 - 1\,\mu_2, \omega_1 = 0, \lambda_2 = 9\,\mu_2\,\omega_2, \lambda_1 = 2\,\mu_2\,\omega_2\}, \{\omega_2 = \omega_2, x_2 = 1, x_1 = 0, \lambda_1 = 2\,\omega_2, \mu_1 = 0, \nu = 0, \mu_2 = 1, \omega_1 = \omega_1, \gamma = 7\,\omega_2, \lambda_2 = -5\,\omega_2\}, \{\omega_2 = \omega_2, \mu_1 = 1, x_1 = 0, \mu_2 = 0, x_2 = 0, \nu = 0, \omega_1 = \omega_1, \lambda_2 = -8\,\omega_1, \gamma = 8\,\omega_1, \lambda_1 = 8\,\omega_1\}$

Discarding solutions of the above systems (taking into account that λ_1, λ_2, μ_1, μ_2, $\nu \geq 0$ *and* ω_1, $\omega_2 > 0$, *we have as possible nondominated solutions in the* x *variables:*

$$\{x_1 = 0, x_2 = 0\}, \{x_1 = 1, x_2 = 0\}, \{x_1 = 0, x_2 = 1\}$$

Now, the non-regularity system is:

(34)
$$\begin{aligned}
\lambda_1 (2\,x_1 - 1) + \nu\,(-1 + 3\,x_1^2) + \mu_1\,\omega_1\,(8 + 2\,x_2) + \mu_2\,\omega_2\,(-2 + 4\,x_2) &= 0\\
\lambda_2 (2\,x_2 - 1) + 4\,\nu\,x_2^3 + \mu_1\,\omega_1\,(-8 + 2\,x_1) + \mu_2\,\omega_2\,(5 + 4\,x_1) &= 0\\
-\mu_1 - \mu_2 &= 0\\
-x_1 + x_1^3 + x_2^4 &= 0\\
x_1^2 - x_1 &= 0\\
x_2^2 - x_2] &= 0
\end{aligned}$$

The set of solution of this system is: $\{\ \{\mu_1 = -\mu_2, \lambda_2 = 6\,\omega_1\,\mu_2 + 9\,\mu_2\,\omega_2, \omega_2 = \omega_2, \lambda_1 = 8\,\omega_1\,\mu_2 - 2\,\nu_1 + 2\,\mu_2\,\omega_2, x_1 = 1, \nu_1 = \nu_1, \mu_2 = \mu_2, \omega_1 = \omega_1, \gamma = \gamma, x_2 = 0\}$, $\{\mu_1 = -\mu_2, \lambda_2 = 8\,\omega_1\,\mu_2 + 5\,\mu_2\,\omega_2, \lambda_1 = -\nu_1 - 2\,\mu_2\,\omega_2 - 8\,\omega_1\,\mu_2, \omega_2 = \omega_2, \nu_1 = \nu_1, \mu_2 = \mu_2, \omega_1 = \omega_1, x_1 = 0, \gamma = \gamma, x_2 = 0\}\ \}$,
whose solutions in the x variables are:

$$\{x_1 = 1, x_2 = 0\}, \{x_1 = 0, x_2 = 0\}$$

Then, the set of possible solutions of our problem is:

$$\{x_1 = 0, x_2 = 0\}, \{x_1 = 1, x_2 = 0\}, \{x_1 = 0, x_2 = 1\}$$

If we try to discard dominated solutions we get that the three solutions are not comparable, and then, all of them are nondominated.

Figure 5.2.1 show the feasible region, the nondominated solutions and the level curves for each of them in this simple problem.

5.2.2. The Chebyshev-Fritz-John approach. Analogously to the previous approach, once we have scalarized the original multiobjective problem to a family of single-objective problems, in this section we apply the Fritz-John (FJ) conditions to all the problems in this family. The following well-known result justifies the use of FJ conditions to obtain candidates to optimal solutions for single-objective problems. Proofs and further details can be found in [10].

THEOREM 5.2.4 (FJ necessary conditions). *Consider the problem:*

(35)
$$\begin{array}{lrll}
\min & f(x) & & \\
s.t. & g_j(x) & \leq 0 & = 1, \ldots, m\\
 & h_r(x) & = 0 & r = 1, \ldots, s\\
 & x & \in \mathbb{R}^n &
\end{array}$$

Let x^* be a feasible solution, and let $J = \{j : g_j(x^*) = 0\}$. Suppose that f and g_j, for $j = 1, \ldots, m$, are differentiable at x^*, and that h_r, for $r = 1, \ldots, s$, is continuously differentiable at x^*. If x^* locally solves Problem (35), then there

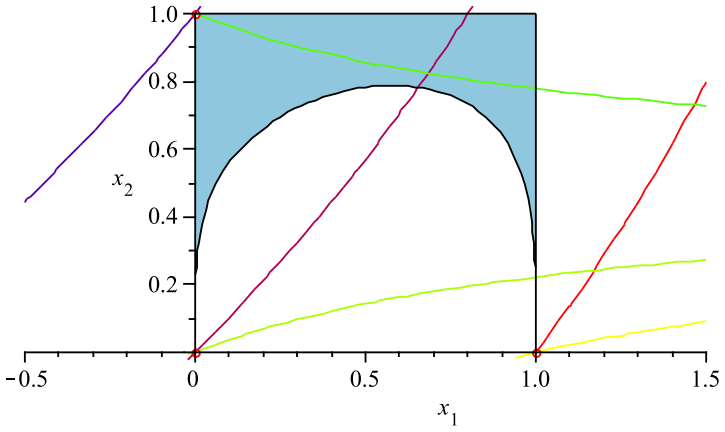


FIGURE 5.3. Feasible region, the nondominated solutions and the level curves of Example 5.2.1

exist scalars λ_j*, for* $j = 1, \ldots, m$*, and* μ_r*, for* $r = 1, \ldots, s$*, such that*

$$\text{(FJ)} \qquad \begin{aligned} \lambda_0 \nabla f(x^*) + \sum_{j=1}^{m} \lambda_j \nabla g_j(x^*) + \sum_{r=1}^{s} \mu_r \nabla h_r(x^*) &= 0 \\ \lambda_j \, g_j(x^*) &= 0 \qquad \text{for } j = 1, \ldots, m \\ \lambda_j &\geq 0 \qquad \text{for } j = 1, \ldots, m \\ (\lambda_0, \lambda, \mu) &\neq (0, \mathbf{0}, \mathbf{0}) \end{aligned}$$

Note that, in the FJ conditions, regularity conditions are not required to set the result.

COROLLARY 5.2.2. *Let* x^* *be a nondominated solution for* $MOPBP_{\mathbf{f},\mathbf{g},\mathbf{h}}$*. Then,* x^* *is a solution of the system of polynomial equations* (36) *for some* $\nu_i, \lambda_j, \mu_r, \beta_l$*, for* $i = 1, \ldots, k, l = 1, \ldots, n$*,* $j = 1, \ldots, m$*,* $r = 1, \ldots, s$ *and*

$\omega > 0$.

$$\left.\begin{array}{r} \lambda_0 - \sum_{i=1}^{k} \nu_i = 0 \\ \sum_{i=1}^{k} \nu_i \omega_i \nabla f_i(x) + \sum_{j=1}^{m} \lambda_j \nabla g_j(x) + \sum_{r=1}^{s} \mu_j \nabla h_r(x) + \sum_{l=1}^{n} \beta_l \, \delta_{il}(2x_i - 1) = 0 \\ \nu_i \left(\omega_i \left(f_i(x) - \hat{y}_i\right) - \gamma\right) = 0 \\ i = 1, \ldots, k \\ \lambda_j \, g_j(x) = 0 \\ j = 0, \ldots, m \end{array}\right\} \tag{36}$$

where $\lambda_j \geq 0$, *for* $j = 1, \ldots, m$, $\nu_i \geq 0$, *for* $i = 1, \ldots, k$ *and* δ_{li}, *for* $l, i = 1, \ldots, n$, *denotes the Kronecker delta function,* $\delta_{li} = \begin{cases} 1 & \text{if } l = i \\ 0 & \text{otherwise} \end{cases}$.

Let X_E^{FJ} denote the set of solutions, in the x-variables, that are feasible solutions of $MOPBP_{\mathbf{f},\mathbf{g},\mathbf{h}}$ and solutions of system (36).

The set of nondominated solutions of our problem is a subset of X_E^{FJ}, since every nondominated solution is an optimal solution for some problem in the form of (36), and every solution of this single-objective problem is a solution of the FJ system.

However, dominated solutions may appear in the set of solutions of (36), so, a final elimination process is to be performed to select only the nondominated solutions.

The steps to solve $MOPBP_{\mathbf{f},\mathbf{g},\mathbf{h}}$ using the Chebyshev-FJ approach are summarized in Algorithm 19.

Algorithm 19: Summary of the procedure for solving MOPBP using the Chebyshev-FJ approach.

Input : $f_1, \ldots, f_k, g_1, \ldots g_m, h_1, \ldots, h_r \in \mathbb{R}[x_1, \ldots, x_n]$

Algorithm:

Step 1: Formulate the Chebyshev scalarization of $MOPBP_{\mathbf{f},\mathbf{g},\mathbf{h}}$. (Problem P_ω)

Step 2: Solve system (36) in the x-variables for any value of $\omega > 0$: X_E^{FJ}.

Step 3: Remove from X_E^{FJ} the set of dominated solutions: X_E.

Output: X_E the set of nondominated solutions for Problem $MOPBP_{\mathbf{f},\mathbf{g},\mathbf{h}}$

THEOREM 5.2.5. *Algorithm 19 solves* $MOPBP_{\mathbf{f},\mathbf{g},\mathbf{h}}$ *in a finite number of steps.*

The last part of the section is devoted to show how to solve the Chebyshev-FJ system using Gröbner bases.

Consider the following polynomial ideal

$$I = \langle \lambda_0 - \sum_{i=1}^{k} \nu_i, \sum_{i=1}^{k} \nu_i \omega_i \nabla f_i(x) + \sum_{j=1}^{m} \lambda_j \nabla g_j(x) + \sum_{r=1}^{s} \mu_j \nabla h_r(x) + \sum_{l=1}^{n} \beta_l \, \delta_{il}(2x_i - 1), \nu_1 \left(\omega_1 \left(f_1(x) - \hat{y}_1\right) - \gamma\right), \ldots, \nu_k \left(\omega_k \left(f_k(x) - \hat{y}_k\right) - \gamma\right), \lambda_1 \, g_1(x), \ldots, \lambda_m \, g_m(x) \rangle$$

in the polynomial ring $\mathbb{R}[\mathbf{x}, \gamma, \lambda, \nu, \mu, \beta]$.

Let us consider a lexicographical order over the monomials in $\mathbb{R}[\mathbf{x}, \gamma, \lambda, \nu, \mu, \beta]$ such that $\mathbf{x} \prec \gamma \prec \lambda \prec \nu \prec \mu \prec \beta$. Then, the Gröbner basis, $\mathcal{G}$, for I with this order has the following triangular shape:

- $\mathcal{G}$ contains one polynomial in $\mathbb{R}[x_n]$: $p_n(x_n)$
- $\mathcal{G}$ contains one or several polynomials in $\mathbb{R}[x_{n-1}, x_n]$: $p_{n-1}^1(x_{n-1}, x_n), \ldots, p_{n-1}^{m_1}(x_{n-1}, x_n)$

...

- $\mathcal{G}$ contains one or several polynomials in $\mathbb{R}[\mathbf{x}]$: $p_1^1(x_1, \ldots, x_n), \ldots, p_1^{m_n}(x_1, \ldots, x_n)$
- The remainder polynomials involve variables $\mathbf{x}$ and at least one of γ, λ, μ, ν or β.

We are interested in finding only the values for the x-variables, so, we avoid the polynomials in $\mathcal{G}$ that involve any of the other auxiliary variables. We denote by $\mathcal{G}^x$ the subset of $\mathcal{G}$ that contains only all the polynomials in the x-variables. By the Extension Theorem, $\mathcal{G}^x$ is a Gröbner basis for $I \cap \mathbb{R}[x_1, \ldots, x_n]$.

Solving the system given by $\mathcal{G}^x$, we obtain as solutions, those of our FJ original system.

The following example illustrates the how the algorithm works.

EXAMPLE 5.2.2. *Consider the following simple problem in two variables:*

$$\begin{array}{lll} \min & (-4\,x_1 + 10\,x_2, -x_1 - 4\,x_2) \\ s.t. & 4\,x_2 - 132\,x_1^4 + 143\,x_1^3 - 22\,x_1 \geq 0 \\ & x_1, x_2 \quad \in \{0, 1\} \end{array} \tag{37}$$

After formulating the Chebyshev problem, the FJ system is:

$$
\begin{aligned}
\lambda_1\,(2\,x_1-1)-4\,\mu_1\,\omega_1-\mu_2\,\omega_2-\lambda_0\,(-528\,x_1^3+429\,x_1^2-22) &= 0,\\
\lambda_2\,(2\,x_2-1)+10\,\mu_1\,\omega_1-4\,\mu_2\,\omega_2-4\,\lambda_0 &= 0,\\
nu-\mu_1-\mu_2 &= 0,\\
x_1^2-x_1 &= 0,\\
x_2^2-x_2 &= 0,\\
\lambda_0\,(4\,x_2-132\,x_1^4+143\,x_1^3-22\,x_1) &= 0,\\
\mu_1\,(\omega_1\,(-4\,x_1+10\,x_2+4)-\gamma) &= 0,\\
\mu_2\,(\omega_2\,(-x_1-4\,x_2+5)-\gamma) &= 0.
\end{aligned}
\tag{38}
$$

Whose solutions using the convenient Gröbner basis are: $\mathcal{G}=\{\{\mu_2=\mu_2,\omega_1=\omega_1,\lambda_1=-4\,\omega_1\,\mu_2+4\,\omega_1\,\nu,\nu=\nu,x_1=1,\lambda_0=0,x_2=0,\gamma=0,\mu_1=\nu-1\,\mu_2,\omega_2=0,\lambda_2=-10\,\omega_1\,\mu_2+10\,\omega_1\,\nu\},\{\mu_1=\nu,\omega_1=\omega_1,\omega_2=\omega_2,\nu=\nu,x_1=1,x_2=1,\lambda_2=-10\,\omega_1\,\nu,\mu_2=0,\lambda_1=4\,\omega_1\,\nu,\gamma=10\,\omega_1,\lambda_0=0\},\{\mu_2=\nu,\omega_1=\omega_1,\omega_2=\omega_2,\nu=\nu,x_2=1,\lambda_0=0,\lambda_2=4\,\omega_2\,\nu,\lambda_1=-1\,\omega_2\,\nu,x_1=0,\mu_1=0,\gamma=\omega_2\},\{\mu_2=\mu_2,\nu=\mu_1+\mu_2,\lambda_1=\mu_2\,\omega_2,\mu_1=\mu_1,\omega_2=\omega_2,x_1=1,x_2=1,\lambda_0=0,\omega_1=0,\lambda_2=4\,\mu_2\,\omega_2,\gamma=0\},\{\mu_1=\nu,\omega_1=\omega_1,\omega_2=\omega_2,\nu=\nu,\mu_2=0,\lambda_0=0,x_1=0,\gamma=4\,\omega_1,x_2=0,\lambda_2=10\,\omega_1\,\nu,\lambda_1=-4\,\omega_1\,\nu\},\{\mu_1=\nu,\lambda_2=\lambda_2,\omega_1=\omega_1,\omega_2=\omega_2,\nu=\nu,\mu_2=0,x_1=0,\gamma=4\,\omega_1,\lambda_0=-0.25\,\lambda_2+2.5\,\omega_1\,\nu,\lambda_1=-5.5\,\lambda_2+51.\,\omega_1\,\nu,x_2=0.\},\{\lambda_0=\lambda_0,\mu_2=\mu_2,\omega_2=\omega_2,\nu=\nu,\lambda_1=-5.\,\omega_2\,\nu+22\,\lambda_0+4\,\mu_2\,\omega_2,x_1=0,x_2=0,\mu_1=\nu-1\,\mu_2,\gamma=5\,\omega_2,\omega_1=1.25\,\omega_2,\lambda_2=-4\,\lambda_0-16.5\,\mu_2\,\omega_2+12.5\,\omega_2\,\nu\},\{\mu_2=\mu_2,\omega_2=\omega_2,\nu=\nu,\lambda_0=0,x_1=0,x_2=0,\mu_1=\nu-1\,\mu_2,\lambda_1=4\,\mu_2\,\omega_2-5\,\omega_2\,\nu,\gamma=5\,\omega_2,\omega_1=1.25\,\omega_2,\lambda_2=-16.5\,\mu_2\,\omega_2+12.5\,\omega_2\,\nu\},\{\mu_2=\nu,\omega_1=\omega_1,\omega_2=\omega_2,\lambda_1=\omega_2\,\nu,\nu=\nu,x_1=1,\lambda_0=0,\mu_1=0,x_2=0,\gamma=4\,\omega_2,\lambda_2=-4\,\omega_2\,\nu\},\{\mu_2=\nu,\lambda_2=\lambda_2,\omega_1=\omega_1,\omega_2=\omega_2,\nu=\nu,x_1=0,\mu_1=0,x_2=0,\gamma=5\,\omega_2,\lambda_0=-.25\,\lambda_2-1\,\omega_2\,\nu,\lambda_1=-5.5\,\lambda_2-23\,\omega_2\,\nu\},\{\mu_2=\mu_2,\omega_2=\omega_2,\nu=\nu,x_2=1.,\lambda_0=0,x_1=0,\mu_1=\nu-1\,\mu_2,\gamma=\omega_2,\omega_1=.07\,\omega_2,\lambda_2=4.71\,\mu_2\,\omega_2-.71\,\omega_2\,\nu,\lambda_1=-.71\,\mu_2\,\omega_2-.28\,\omega_2\,\nu\},\{\mu_2=\nu,\omega_1=\omega_1,\omega_2=\omega_2,\lambda_1=\omega_2\,\nu,\nu=\nu,x_1=1,x_2=1,\lambda_0=0,\lambda_2=4\,\omega_2\,\nu,\mu_1=0,\gamma=0\},\{\gamma=\gamma,\omega_1=\omega_1,\omega_2=\omega_2,x_1=1,\mu_2=0,\lambda_0=0,\mu_1=0,x_2=0,\lambda_1=0,\nu=0,\lambda_2=0\},\{\mu_1=\nu,\omega_1=\omega_1,\omega_2=\omega_2,\nu=\nu,x_1=1,\mu_2=0,\lambda_1=4\,\omega_1\,\nu,\lambda_0=0,x_2=0,\gamma=0,\lambda_2=10\,\omega_1\,\nu\},\{\mu_2=\nu,\omega_1=\omega_1,\omega_2=\omega_2,\nu=\nu,\lambda_0=0,\lambda_1=-1\,\omega_2\,\nu,x_1=0,\mu_1=0,x_2=0,\lambda_2=-4\,\omega_2\,\nu,\gamma=5\,\omega_2\},\{\gamma=\gamma,\omega_1=\omega_1,\omega_2=\omega_2,\mu_2=0,\lambda_0=0,x_1=0,\mu_1=0,x_2=0,\lambda_1=0,\nu=0,\lambda_2=0\},\{\mu_1=\nu,\omega_1=\omega_1,\omega_2=\omega_2,\nu=\nu,x_2=1,\lambda_2=-10\,\omega_1\,\nu,\mu_2=0,\gamma=14\,\omega_1,\lambda_0=0,x_1=0,\lambda_1=-4\,\omega_1\,\nu\},\{\lambda_2=-4\,\lambda_0,\lambda_1=22\,\lambda_0,\gamma=\gamma,\lambda_0=\lambda_0,\omega_1=\omega_1,\omega_2=\omega_2,\mu_2=0,x_1=0,\mu_1=0,x_2=0,\nu=0\},\{\gamma=\gamma,\omega_1=\omega_1,\omega_2=\omega_2,x_1=1,x_2=1,\mu_2=0,\lambda_0=0,\mu_1=0,\lambda_1=0,\nu=0,\lambda_2=0\},\{\gamma=\gamma,\omega_1=$

$\omega_1, \omega_2 = \omega_2, x_2 = 1, \mu_2 = 0, \lambda_0 = 0, x_1 = 0, \mu_1 = 0, \lambda_1 = 0, \nu = 0, \lambda_2 = 0\}\}$
Discarding those solutions that do not verify the FJ inequality conditions we have as possible nondominated solutions:

$$\{x_1 = 0, x_2 = 0\}, \{x_1 = 0, x_2 = 1\}$$

Furthermore, both solutions are not comparable, so they are the nondominated solutions of the problem.

Figure 5.2.2 shows feasible region, the nondominated solutions and the level curves of Example 5.2.2.

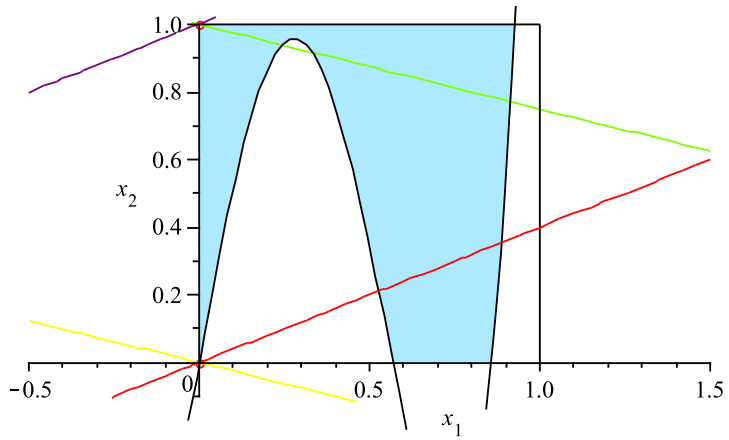


FIGURE 5.4. Feasible region, the nondominated solutions and the level curves of Example 5.2.2

REMARK 5.2.1 (Convex Case). *In the special case where both objective functions and constraints are convex, sufficient KKT conditions can be applied. If the feasible solution x^* satisfies KKT conditions, and all objective and constraints functions are convex, then x^* is a nondominated solution. As a particular case, this situation is applicable to linear problems.*

In this case, we may choose a linear scalarization instead of the Chebyshev scalarization. With this alternative approach, the scalarized problem is

$$\begin{array}{lrll} \min & \sum_{s=1}^{k} t_s\, f_s(x) & & \\ s.t. & g_j(x) & \leq 0 & j = 1, \ldots, m \\ & h_r(x) & = 0 & r = 1, \ldots, s \\ & x_i(x_i - 1) & = 0 & i = 1, \ldots, n \end{array}$$

for $t_1, \ldots, t_k > 0$.

Then, by Corollary 11.19 in [63], and denoting by S *the feasible region, if* $f(S) + \mathbb{R}^k_+$ *is convex, then each* x^* *is a nondominated solution if and only if* x^* *is a solution of Problem 5.2.1 for some* $t_1, \ldots, t_k > 0$.

Using both results, necessary and sufficient conditions are given for that problem and the removing step is avoided.

REMARK 5.2.2 (Single-Objective Case). *The same approach can be applied to solve single-objective problems. In this case, KKT (or FJ) conditions can be applied directly to the original problem, without scalarizations.*

5.3. Obtaining nondominated solutions by nondominance conditions

In this section, we address the solution of $MOPBP_{\mathbf{f},\mathbf{g},\mathbf{h}}$ by directly applying necessary conditions for multiobjective problems. With these conditions we do not need to scalarize the problem, as in the above section, avoiding some steps in the process followed in the previous sections.

The following result states the Fritz-John necessary optimality conditions for multiobjective problems.

THEOREM 5.3.1 (Multiobjective FJ necessary conditions, Theorem 3.1.1. in [81]). *Consider the problem:*

$$\begin{array}{lrll} \min & (f_1(x), \ldots, f_k(x)) & & \\ s.t. & g_j(x) & \leq 0 & = 1, \ldots, m \\ & h_r(x) & = 0 & r = 1, \ldots, s \\ & x & \in \mathbb{R}^n & \end{array} \tag{39}$$

Let x^* *a feasible solution. Suppose that* f_i, *for* $i = 1, \ldots, k$, g_j, *for* $j = 1, \ldots, m$ *and* h_r, *for* $r = 1, \ldots, s$, *are continuously differentiable at* x^*. *If* x^* *is a nondominated solution for Problem 39, then there exist scalars* ν_i, *for* $i = 1, \ldots, k$, λ_j, *for* $j = 1, \ldots, m$, *and* μ_r, *for* $r = 1, \ldots, s$, *such that*

(MO-FJ)

$$\begin{aligned}
\sum_{i=1}^{k} \nu_i \nabla f_i(x^*) + \sum_{j=1}^{m} \lambda_j \nabla g_j(x^*) + \sum_{r=1}^{s} \mu_r \nabla h_r(x^*) &= 0 \\
\lambda_j \, g_j(x^*) &= 0 \qquad \text{for } j = 1, \ldots, m \\
\lambda_j &\geq 0 \qquad \text{for } j = 1, \ldots, m \\
\nu_i &\geq 0 \qquad \text{for } i = 1, \ldots, k \\
(\nu, \lambda, \mu) &\neq (\mathbf{0}, \mathbf{0}, \mathbf{0})
\end{aligned}$$

With this result, one can solve the system given by the necessary conditions to obtain candidates to be nondominated solutions for our problem. For solving this system, we use lexicographical Gröbner bases as in the above sections. We summarize the algorithm for solving the multiobjective polynomial problem in Algorithm 20.

Algorithm 20: Summary of the procedure for solving MOPBP using the multiobjective FJ optimality conditions.

Input : $f_1, \ldots, f_k, g_1, \ldots g_m, h_1, \ldots, h_r \in \mathbb{R}[x_1, \ldots, x_n]$

Algorithm:

Step 1: Solve system (MO-FJ): X_E^{MOFJ}.

Step 2: Remove from X_E^{MOFJ} the subset of dominated solutions: X_E.

Output: X_E the set of nondominated solutions for Problem $MOPBP_{\mathbf{f},\mathbf{g},\mathbf{h}}$

The following simple example illustrates Algorithm 20.

Example 5.3.1.

$$\begin{aligned}
&\min && (x_1^2 + 10x_2, -4x_1 - x_2^3) \\
&s.t. && 16 * x_2 - 16 * x_1^3 + 16 * x_1^2 - 3x_1 - 8 \geq 0 \\
& && x_1, x_2 \in \{0, 1\}
\end{aligned} \tag{40}$$

Figure 5.5 shows the feasible region of the problem.

The system after applying Theorem 5.3.1 to the problem is:

$$\begin{aligned}
2\,\nu_1\,x_1 - 4\,\nu_2 + \lambda_1\,(2\,x_1 - 1) - \mu_1\,(-3\,x_1^2 + 2\,x_1 - 3/16) &= 0 \\
10\,\nu_1 - 3\,\nu_2\,x_2^2 + \lambda_2\,(2\,x_2 - 1) - \mu_1 &= 0 \\
\mu_1\,(x_2 - x_1^3 + x_1^2 - (3/16)\,x_1 - 1/2) &= 0 \\
x_1^2 - x_1 &= 0 \\
x_2^2 - x_2 &= 0 \\
\lambda_1, \lambda_2 &\geq 0 \\
\nu_1, \nu_2 &\geq 0
\end{aligned} \tag{41}$$

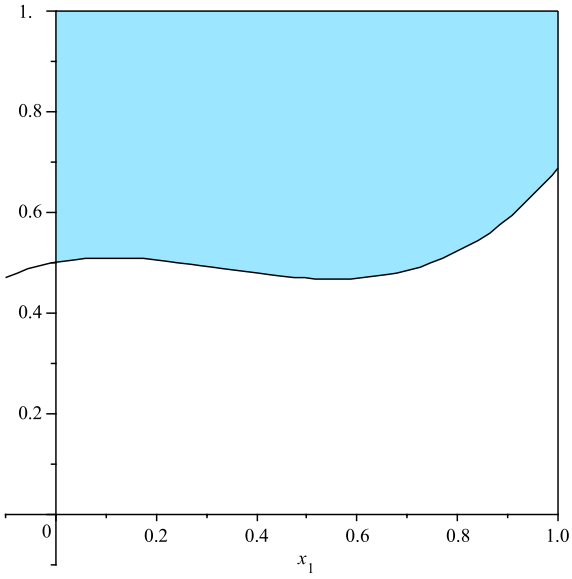


FIGURE 5.5. Feasible region of Example 5.3.1

whose solutions are:

$\{x_2 = 0, \mu_1 = 0, x_1 = 1, \nu_1 = \nu_1, \nu_2 = \nu_2, \lambda_2 = 10\,\nu_1, \lambda_1 = -2\,\nu_1 + 4\,\nu_2\}$,

$\{x_2 = 1, \mu_1 = 0, x_1 = 1, \nu_1 = \nu_1, \nu_2 = \nu_2, \lambda_2 = 3\,\nu_2 - 10\,\nu_1, \lambda_1 = -2\,\nu_1 + 4\,\nu_2\}\}$.

Discarding dominated solutions we obtain that the set of nondominated solutions is:

$$\{x_1 = 1, x_2 = 1\}, \{x_1 = 0, x_2 = 1\}$$

Figure 5.6 shows this set of nondominated solutions in the feasible region and the level curves at these points of the problem.

REMARK 5.3.1. *In the special case where both objective functions and constraints are convex, Theorem 5.3.1 gives sufficient nondominance conditions for* $MOPBP_{\mathbf{f},\mathbf{g},\mathbf{h}}$ *requiring that* $\nu_i > 0$ *(see Theorem 3.1.8 in [81]).*

5.4. Computational Experiments

A series of computational experiments have been performed in order to evaluate the behavior of the proposed solution methods. Programs have been coded in MAPLE 11 and executed in a PC with an Intel Core 2 Quad processor at 2x 2.50 Ghz and 4 GB of RAM. The implementation has been done in that symbolic programming language, available upon request, in order to make the access easy to both optimizers and algebraic geometers.

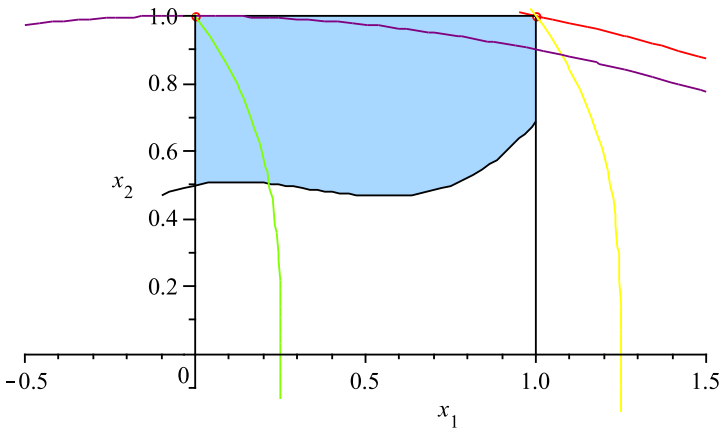


FIGURE 5.6. Nondominated solutions and level curves of Example 5.3.1

We run the algorithms for three families of binary biobjective and triobjective knapsack problems: linear, quadratic and cubic, and for a biobjective portfolio selection model. For each problem, we obtain the set of nondominated solutions as well as the CPU times for computing the corresponding Gröbner bases associated to the problems, and the total CPU times for obtaining the set of solutions.

We give a short description of the problems where we test the algorithms. In all cases, we use binary variables x_j, $j = 1, \ldots, n$, where $x_j = 1$ means that the item (resp. security) j is selected for the knapsack (resp. portfolio) problem.

(1) *Biobjective (linear) knapsack problem* (`biobj_linkn`): Assume that n items are given. Item j has associated costs q_j^1, q_j^2 for two different targets, and a unit profit a_j, $j = 1, \ldots, n$. The biobjective knapsack problem calls for selecting the item subsets whose overall profit ensures a knapsack with value at least b, so as to minimize

(in the nondominance sense) the overall costs. The problem may be formulated:

$$\min\ (\sum_{j=1}^{n} q_j^1\, x_j, \sum_{j=1}^{n} q_j^2\, x_j)$$
$$s.t. \quad \sum_{i=1}^{n} a_i\, x_i \geq b, \quad x \in \{0,1\}^n$$

(2) *Biobjective cubic knapsack problem* (`biobj_cubkn`): Assume that n items are given where item j has an integer profit a_j. In addition we are given two $n \times n \times n$ matrices $P^1 = (p^1_{ijk})$ and $P^2 = (p^2_{ijk})$, where p^1_{ijk} and p^2_{ijk} are the costs for each of the targets if the combination of items i, j, k is selected for $i < j < k$; and two additional $n \times n$ matrices $Q^1 = (q^1_{ij})$ and $Q^2 = (q^2_{ij})$, where q^1_{jj} and q^2_{ij} are the costs for the two different targets if both items i and j are selected for $i < j$ The biobjective cubic knapsack problem calls for selecting the item subsets whose overall profit exceeds the purpose of the knapsack b, so as to minimize the overall costs. The problem may be formulated:

$$\min\ (\sum_{i=1}^{n}\sum_{j=i}^{n} q^1_{ij}\, x_i\, x_j + \sum_{i=1}^{n-2}\sum_{j=i+1}^{n-1}\sum_{l=j+1}^{n} p^1_{ijl}\, x_i\, x_j\, x_l,$$
$$\sum_{i=1}^{n}\sum_{j=i}^{n} q^2_{ij}\, x_i\, x_j + \sum_{i=1}^{n-2}\sum_{j=i+1}^{n-1}\sum_{l=j+1}^{n} p^2_{ijl}\, x_i\, x_j\, x_l)$$
$$s.t. \quad \sum_{i=1}^{n} a_i\, x_i \geq b, \quad x \in \{0,1\}^n$$

(3) *Biobjective quadratic knapsack problem* (`biobj_qkn`): This problem may be seen as a special case of the biobjective cubic knapsack problem when there are no cost correlations between triplets.

(4) *Triobjective (linear) knapsack problem* (`triobj_linkn`): Assume that n items are given where item j has an integer profit a_j. In addition, we are given three vectors $q^1 = (q^1_j)$, $q^2 = (q^2_j)$ and $q^3 = (q^3_j)$, where q^1_j,q^2_j and q^3_j are the costs for three different targets if j is selected. The triobjective knapsack problem calls for selecting the item subsets whose overall profit ensures a profit for the knapsack at least b, so as to minimize (in the nondominance sense) the overall costs. The problem is:

$$\min\ (\sum_{j=1}^{n} q_j^1\, x_j, \sum_{j=1}^{n} q_j^2\, x_j, \sum_{j=1}^{n} q_j^3\, x_j)$$
$$s.t. \sum_{i=1}^{n} a_i\, x_i \geq b, \quad x \in \{0,1\}^n$$

(5) *Triobjective cubic knapsack problem* (`triobj_cubkn`): We are given n items where item j has an integer profit a_j. In addition, we are given three $n \times n \times n$ matrices $P^1 = (p^1_{ijk}), P^2 = (p^2_{ijk})$ and $P^3 = (p^3_{ijk})$, where p^1_{ijk}, p^2_{ijk} and p^3_{ijk} are the costs for each of the targets if the combination of items i, j and k is selected for $i < j < k$; and three additional $n \times n$ matrices $Q^1 = (q^1_{ij}), Q^2 = (q^2_{ij})$ and $Q^3 = (q^2_{ij})$, where q^1_{jj}, q^2_{ij} and q^3_{ij} are the costs for three different targets if both items i and j are selected for $i < j$ The triobjective cubic knapsack problem calls for selecting the item subsets whose overall profit ensures a value of b, so as to minimize the overall costs. The problem is:

$$\begin{aligned}\min \quad & \Big(\sum_{i=1}^{n}\sum_{j=i}^{n} q^1_{ij}\, x_i\, x_j + \sum_{i=1}^{n-2}\sum_{j=i+1}^{n-1}\sum_{l=j+1}^{n} p^1_{ijl}\, x_i\, x_j\, x_l, \\ & \sum_{i=1}^{n}\sum_{j=i}^{n} q^2_{ij}\, x_i\, x_j + \sum_{i=1}^{n-2}\sum_{j=i+1}^{n-1}\sum_{l=j+1}^{n} p^2_{ijl}\, x_i\, x_j\, x_l, \\ & \sum_{i=1}^{n}\sum_{j=i}^{n} q^3_{ij}\, x_i\, x_j + \sum_{i=1}^{n-2}\sum_{j=i+1}^{n-1}\sum_{l=j+1}^{n} p^3_{ijl}\, x_i\, x_j\, x_l\Big) \\ & s.t. \sum_{i=1}^{n} a_i\, x_i \geq b, \quad x \in \{0,1\}^n\end{aligned}$$

(6) *Triobjective quadratic knapsack problem* (`triobj_qkn`): This problem may be seen as a special case of the triobjective cubic knapsack problem when there are no cost correlations between triplets.

(7) *Biobjective portfolio selection* (`portfolio`): Consider a market with n securities. An investor with initial wealth b seeks to improve his wealth status by investing it into these n risky securities. Let X_i be the random return per a lot of the i-th secutiry ($i = 1, \ldots, n$). The mean, $\mu_i = E[X_i]$, and the covariance, $\sigma_{ij} = Cov(X_i, X_j), i, j = 1, \ldots, n$, of the returns are assumed to be known. Let x_i be a decision variable that takes value 1 if the decision-maker invests in the i-th security and 0 otherwise. Denote the decision vector by $x = (x_1, \ldots, x_n)$. Then, the random return for a inversion vector x from the securities is $\sum_{i=1} x_i X_i$ and the mean and variance of this random variable are $E[\sum_{i=1} x_i X_i] = \sum_{i=1}^{n} \mu_i x_i$ and $Var(\sum_{i=1} x_i X_i) = \sum_{i=1}^{n}\sum_{j=1}^{n} x_i\, x_j\, \sigma_{ij}$.

Let a_i be the current price of the i-th security. Then, if an investor looks for minimizing his investment risk and simultaneously maximizing the expected return with that investment, the problem can be formulated as:

$$\left.\begin{array}{l} \min\ (Var(\sum_{i=1}^{n} x_i\, X_i), -E[\sum_{i=1}^{n} x_i\, X_i]) \\ s.t. \sum_{i=1}^{n} a_i\, x_i \leq b, \quad x \in \{0,1\}^n \end{array}\right\} \quad \Rightarrow \quad \left.\begin{array}{l} \min\ (\sum_{i=1}^{n} \sigma_{ij}\, x_i\, x_j, -\sum_{i=1}^{n} \mu_i\, x_i) \\ s.t. \sum_{i=1}^{n} a_i\, x_i \leq b, \quad x \in \{0,1\}^n \end{array}\right\}$$

For each of the above 7 classes of problems, we consider instances randomly generated as follows: a_i is randomly drawn in $[-10, 10]$ and the coefficients of the objective functions, q_{ij}^k, p_{ijl}^k, σ_{ij} and μ_i, range in $[-10, 10]$. Once the constraint vector, $(a_1, \ldots, a_n)$, is generated, the right hand side, b, is randomly generated in $[1, |\sum_{i=1}^{n} a_i|]$. For each type of instances and each value of n in $[2, 13]$ we generated 5 instances.

Tables 5.1 and 5.3 contain a summary of the average results over the different instances generated for the above problems. Each algorithm is labeled conveniently: `alg1` corresponds with Algorithm 17, `kkt` is Algorithm 18, `kkt_sl` is Algorithm 18 where the inequality is transformed to an equation using a slack variable, `fj` is Algorithm 19, `fj_sl` is Algorithm 19 where the inequality is transformed to an equation using a slack variable and `mofj` stands for Algorithm 20. For each of these algorithms we present the CPU time for computing the corresponding Gröbner basis (`tgb`), the total CPU time for obtaining the set of nondominated solutions (`ttot`), the number of nondominated solutions (`#nd`) and the number of variables involved in the resolution of the problem (`#vars`).

From those tables, the reader may note that Algorithm 17 is faster than the others for the smallest instances, although the CPU times for this algorithm increase faster than for the others and it is not able to obtain solutions when the size of the problem is around 12 variables. The algorithms based on Chebyshev scalarization (`kkt`, `kkt_sl`, `fj` and `fj_sl`) are better than `alg1` for the largest instances. The differences between these four methods are meaningful, but the algorithms based on the KKT conditions are, in almost all the instances, faster than those based on the FJ conditions. Note that considering slack variables to avoid the inequality constraint is not better, since the CPU times when the slack variable is considered are larger. Finally, the best algorithm, in CPU time, is `mofj` since except for the small instances is the fastest and it was able to solve larger instances.

One may think that the last step of our methods, i.e. removing dominated solutions, should be more time consuming in `alg1` than in the remaining methods since `alg1` does not use optimality conditions. However, from our experiments this conclusion is not clearly supported. Actually, although this process, is time consuming, when the dimension of the problem increases this time is rather small compared with the effort necessary to obtain the Gröbner bases.

Analyzing the CPU times for computing the Gröbner basis for each procedure and comparing with the total time, we can obtain, approximately, the time consumed to remove dominated solutions to finally obtain the nondominated ones. For `alg1`, it was the procedure consuming the biggest average part of the time removing the dominated solutions for biobjective linear knapsack problems (% 41.13), triobjective linear knapsack (% 42.68) and triobjective knapsack. However, the coefficient of variation for these distribution (percentage of the total time consumed computing the Gröbner basis) for `alg1` is in all cases around 40%, while for the others algorithms this coefficient is always around 8%, so, the relative time consumed computing the reduced Gröbner basis varies more than in the other methods. Actually, for example, for biobjective linear knapsack problems it ranges for `alg1` in $[0.26, 0.92]$ while in `kkt` ranges in $[0.58, 0.65]$, in `fj` ranges in $[0.62, 0.69]$ and in `mofj` ranges in $[0.54, 0.66]$. Furthermore, in general, for `alg1`, the percentage of the total time that is consumed computing the Gröbner basis increases when the dimension of the problem is higher. This difference is not so significative for the other methods. Then, it is clear, that although the cleaning process in `alg1` (that means selecting the efficient solutions) seems to be the hardest, increasing the number of variables one can observe that a big part of the time used on computing the Gröbner basis, and only a small part of it on discarding the componentwise dominated solutions.

Figure 5.4 shows some graphics comparing the algorithms with respect to the average coefficient of variation of the time consumed removing dominated elements for any of the problems and algorithms.

Table 5.2 shows some information about each of the presented algorithms. For a multiobjective problem with n variables, m polynomial inequality constraints given by $\mathbf{g} = (g_1, \ldots, g_m)$, s polynomial equality constraints given by $\mathbf{h} = (h_1, \ldots, h_s)$ and k objectives functions given by $\mathbf{f} = (f_1, \ldots, f_k)$, Table 3 shows the number of variables (`#var`), the number of generators (`#gen`) and the maximal degrees (`maxdeg`) of the initial polynomial ideals related to the each of the algorithms. These numbers inform us about the theoretical complexity of the algorithms. The computation of Gröbner bases depends of the

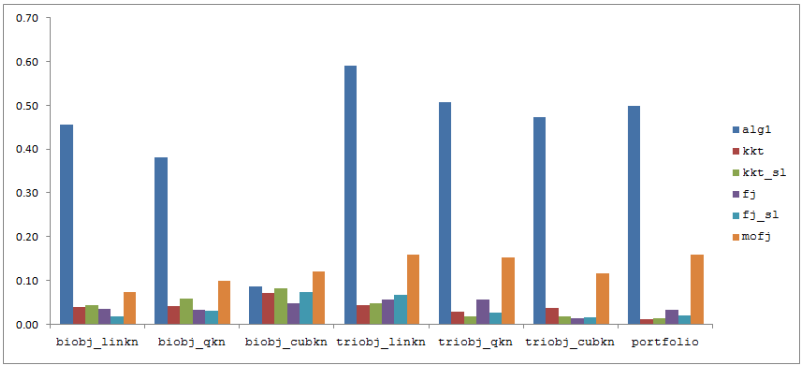


FIGURE 5.7. Graph comparing the coefficients of variation for the time consumed in the removing process for all the algorithms and problems.

number of variables (in general, double exponential) and of the size of the initial system of generators (degrees and number of polynomials). Actually, it is known that computing a Gröbner basis using Buchberger Algorithm is doubly exponential in the number of variables. Some complexity bounds for this algorithm involving `#var`, `#gen` and `maxdeg` can be found in [39].

From the above table the reader may note that both `alg1` and `mojf` have the same number of variables in any case, but the number of initial generators for `alg1` is, in general, smaller than the same number for `mojf`, since the number of objectives is usually smaller than the number of variables. Furthermore, maximal degrees are smaller in `alg1` than in `mojf` . However, in practice, `mojf` is faster than `alg1` since using optimality conditions helps in identifying nondominated solutions.

		alg1			kkt			kkt_sl			fj			fj_sl			mofj			
prob	n	gbt	tott	#vars	gbt	tott	#vars	gbt	tott	#vars	gbt	tott	#vars	gbt	tott	#vars	gbt	tott	#vars	#nd
biobj_linkn	2	0.02	0.07	5	0.30	0.48	11	0.35	0.61	12	0.24	0.39	10	0.31	0.50	11	0.03	0.06	7	1.8
	3	0.03	0.09	6	0.82	1.31	13	0.95	1.61	14	0.71	1.06	12	0.84	1.39	13	0.10	0.18	9	1.6
	4	0.07	0.23	7	2.45	3.89	15	3.05	4.90	16	2.07	3.10	14	2.54	4.12	15	0.33	0.52	11	3.6
	5	0.32	0.69	8	7.01	10.82	17	8.71	13.90	18	5.92	8.71	16	7.31	11.71	17	0.91	1.40	13	4.6
	6	2.86	4.11	9	25.00	37.88	19	32.29	50.13	20	20.63	30.11	18	26.66	41.70	19	3.01	4.61	15	5.4
	7	31.15	35.59	10	72.85	107.15	21	82.90	127.08	22	55.68	79.80	20	66.77	104.82	21	7.89	11.60	17	5
	8	342.10	373.72	11	176.94	261.58	23	209.68	322.58	24	133.78	193.54	22	165.97	262.46	23	18.35	27.60	19	4.2
	9	5273.56	6382.43	12	462.14	675.24	25	529.50	813.54	26	333.81	492.89	24	418.75	670.82	25	48.50	79.08	21	8
	10																269.19	404.04	23	7.8
	11																480.26	835.46	25	6.4
	12																1340.31	2004.33	27	5.4
	13																4091.92	19546.05	29	11
biobj_qkn	2	0.03	0.07	5	0.28	0.47	11	0.37	0.64	12	0.23	0.36	10	0.31	0.51	11	0.04	0.07	7	1.4
	3	0.04	0.10	6	0.79	1.29	13	1.12	1.86	14	0.68	1.04	12	0.95	1.54	13	0.09	0.16	9	2
	4	0.19	0.37	7	3.27	4.97	15	4.63	7.15	16	2.69	4.01	14	3.89	5.95	15	0.49	0.70	11	2.4
	5	1.76	2.37	8	9.22	13.88	17	11.94	18.06	18	7.54	10.94	16	9.90	15.11	17	1.08	1.74	13	3.8
	6	21.43	23.22	9	25.21	37.33	19	33.74	50.13	20	20.57	29.46	18	27.60	41.86	19	2.92	4.58	15	3.6
	7	425.13	430.43	10	59.57	91.29	21	85.56	129.88	22	49.38	72.81	20	69.20	107.76	21	5.93	9.71	17	4.4
	8				172.26	255.22	23	228.66	337.02	24	127.43	188.32	22	174.23	272.72	23	14.75	25.66	19	5.4
	9				463.39	692.25	25	641.29	939.99	26	350.34	515.07	24	489.31	755.88	25	39.90	65.78	21	6.6
	10																138.11	255.42	23	7
	11																331.34	643.90	25	9
	12																891.46	1833.75	27	8
biobj_cubkn	3	0.17	0.21	7	0.72	1.14	13	0.97	1.56	14	0.62	0.92	12	0.82	1.31	13	0.09	0.15	9	1.8
	4	1.90	2.00	8	4.68	6.73	15	6.26	8.98	16	3.86	5.37	14	5.12	7.38	15	0.56	0.83	11	1.8
	5	7.26	7.48	9	9.29	13.43	17	12.49	17.79	18	7.86	10.81	16	10.31	14.91	17	1.11	1.65	13	2.8
	6	205.52	206.25	10													6.11	8.29	15	2.2
	7	2067.24	2069.38	11													11.25	15.24	17	6.6
	8																24.98	32.25	19	5.2
	9																81.80	106.90	21	6.2
	10																258.68	389.95	23	5.8
	11																690.43	916.80	25	7.8

TABLE 5.1. Computational results for biobjective knapsack problems.

Algorithm	#var	#gen	maxdeg
alg1	$2n+k+m+s$	$n+k+m+s$	$\max\{2, deg(f), deg(g), deg(h)\}$
kkt	$2n+2k+m+s+1$	$2n+k+m+s+1$	$\max\{deg(f)+2, deg(g)+1, deg(h)\}$
nr	$2n+2k+m+s+1$	$2n+m+s$	$\max\{deg(f)+1, deg(g), deg(h)\}$
fj	$2n+2k+m+s+2$	$2n+k+m+s+1$	$\max\{deg(f)+2, deg(g)+1, deg(h)\}$
mojf	$2n+k+m+s$	$2n+m+s$	$\max\{deg(f), deg(g)+1, deg(h)\}$

TABLE 5.2. Some data about all the nonlinear approaches.

		alg1			kkt			kkt_sl			fj			fj_sl			mofj			
prob	n	gbt	tott	#var	gbt	tott	#var	gbt	tott	#var	gbt	tott	#var	gbt	tott	#var	gbt	tott	#var	#nd
triobj_linkn	2	0.03	0.08	6	0.82	1.27	13	0.99	1.61	14	0.75	1.16	12	0.88	1.47	13	0.04	0.06	8	1.6
	3	0.02	0.10	7	2.98	4.33	15	3.63	5.51	16	4.19	5.62	14	4.95	7.00	15	0.14	0.20	10	2
	4	0.06	0.65	8	14.71	20.07	17	18.03	25.22	18	19.93	25.61	16	23.62	31.94	17	0.73	0.90	12	2.8
	5	0.71	1.00	9	40.75	56.40	19	48.20	68.21	20	55.31	72.04	18	65.38	87.07	19	2.70	3.62	14	4.2
	6	2.44	3.32	10	68.69	102.22	21	84.95	129.12	22	107.04	147.24	20	130.13	185.43	21	2.26	3.36	16	6.2
	7	22.35	25.32	11	188.86	276.15	23	219.78	335.08	24	272.32	374.25	22	326.76	470.51	23	5.89	8.59	18	11.4
	8	360.38	367.76	12	501.35	731.46	25	576.07	856.28	26	729.93	993.07	24	830.08	1194.75	25	17.16	23.83	20	4.4
	9																69.68	115.30	22	24.4
	10																193.15	301.04	24	31.4
	11																529.51	1075.93	26	26
	12																1422.70	3145.30	28	85.2
triobj_qkn	2	0.04	0.09	6	0.96	1.52	13	1.28	1.96	14	0.88	1.37	12	1.19	1.81	13	0.05	0.08	8	2.2
	3	0.06	0.18	7	3.20	4.72	15	4.11	6.13	16	4.67	6.25	14	5.21	7.42	15	0.14	0.20	10	2.8
	4	0.09	0.32	8	6.50	9.89	17	9.03	13.82	18	9.59	13.14	16	11.44	16.59	17	0.28	0.45	12	5.2
	5	2.87	3.68	9	21.05	30.87	19	30.61	45.61	20	30.93	41.79	18	39.81	56.92	19	0.79	1.26	14	8.6
	6	31.76	33.69	10	49.11	72.24	21	72.03	106.16	22	69.31	91.93	20	86.83	122.30	21	2.19	3.85	16	9.4
	7	1099.24	1109.07	11	152.63	224.47	23	223.51	326.03	24	232.64	319.26	22	296.38	417.00	23	5.18	8.14	18	12.6
	8				481.73	709.09	25	721.64	1078.26	26	700.63	980.21	24	904.31	1289.78	25	14.81	22.22	20	17.6
	9																44.36	67.14	22	13.6
	10																119.50	213.66	24	17.8
	11																343.53	793.88	26	29.8
	12																1018.35	2445.64	28	40.2
triobj_cubkn	3	0.05	0.15	7	2.74	4.01	15	4.07	5.88	16	4.25	5.55	14	5.33	7.29	15	0.11	0.19	10	3.4
	4	0.18	0.43	8	9.97	13.81	17	13.14	18.50	18	14.32	18.31	16	17.91	23.90	17	0.42	0.56	12	4.4
	5	1.37	1.82	9													1.03	1.43	14	8.4
	6	28.61	29.80	10													2.93	4.19	16	6
	7																9.57	12.67	18	7.6
	8																30.95	37.29	20	13.4
	9																92.52	115.68	22	25.4
	10																371.98	447.80	24	29.4
	11																1006.29	1593.48	26	37.4
portfolio	2	0.03	0.14	5	0.43	0.73	11	0.68	1.10	12	0.40	0.65	10	0.60	0.94	11	0.06	0.08	7	2.0
	3	0.03	0.08	6	1.40	2.09	13	1.65	2.59	14	1.08	1.64	12	1.42	2.28	13	0.18	0.28	9	2.0
	4	0.08	0.38	7	3.50	5.34	15	4.56	7.08	16	2.85	4.17	14	3.74	5.95	15	0.46	0.71	11	4.0
	5	0.82	1.33	8	9.41	14.46	17	11.87	18.16	18	7.85	11.61	16	9.76	15.15	17	1.16	1.84	13	5.4
	6	17.39	18.73	9	25.28	38.45	19	32.75	50.01	20	20.92	30.43	18	26.76	41.96	19	2.90	4.71	15	6.8
	7	179.84	184.29	10	74.90	111.93	21	95.74	143.34	22	59.19	85.91	20	75.66	117.77	21	7.40	12.09	17	6.2
	8	4739.76	4749.91	11	132.97	201.62	23	168.92	255.92	24	108.75	161.05	22	137.98	217.42	23	15.16	25.23	19	8.0
	9				393.58	606.13	25	580.04	886.54	26	311.33	490.41	24	455.34	742.80	25	35.83	76.77	21	11.4
	10				1212.90	1837.36	27	1601.12	2440.86	28	869.72	1379.88	26	1207.49	1982.07	27	101.17	221.92	23	12.2
	11																305.80	724.28	25	15.6

TABLE 5.3. Computational results for triobjective knapsack and biobjective portfolio problems.

CHAPTER 6

Conclusions

In this thesis we present several methods to solve two different families of multiobjective discrete problems: linear and polynomial integer programs. The linear case is tackled using the new structure that we propose, partial Gröbner bases, in both polynomial and geometrical languages and generating functions. With partial Gröbner bases we give a methodology analogous to the one given by Conti and Traverso, but for multiobjective problems. The advantage of this tool is that the geometrical interpretation is intuitive and its is easy to describe and implement the algorithm to obtain the Pareto-optimal solutions. However, the generating function approach uses Barvinok's ideas for encoding the integer points inside polytopes to encode the Pareto-optimal solutions of a multiobjective linear integer problem. With this tool, we can give some new complexity results, proving that encoding these Pareto-optimal solutions can be done in polynomial time for fixed dimension of the decision space (without fixing the dimension of the space of objectives). These kind of results seem to be impossible to obtain with partial Gröbner bases since even in the single-objective case it is not clear the complexity of Gröbner basis (the known upper bound is exponential in the dimension of the decision space). As opposite to that, an implementation of the best algorithm (with respect to the complexity) using generating functions is not easy. For that, we would need to implement the projection and the intersection operations. A difficult task, since the constructive proofs of that results use many other results from discrete geometry. For partial Gröbner bases, although there are some subroutines, the algorithm to obtain the set of Pareto-optimal solutions is almost self-contained and the implementation is not that difficult (only some thousand hours). Moreover, the geometrical approach of partial Gröbner bases makes intuitive and clear the implementation. Other advantage of p-Gröbner bases is that we have identify some patterns in the Gröbner bases (in the single-objective case) and in the partial Gröbner bases for some well-known combinatorial optimization problems. If a given structure of the partial Gröbner basis is given a priori, obtaining the Pareto optimal solutions of the corresponding problem should not be very expensive. In practice we

show that computing the partial Gröbner basis consumes around 90% of the time. Furthermore, the geometrical interpretation of the p-Gröbner basis of a combinatorial problem coincides with the movements in the graph that "improve" feasible solutions. We have not already identified any pattern of the generating functions encoding the Pareto-optimal solutions since only some small problems could be computed "by hand".

The other family of problems that we have solved is that of multiobjective polynomial integer programs. To solve them, we use the property of lexicographic Gröbner bases for solving systems of polynomial equations. This property is based on the extension and the elimination property. To apply these results, first we transform the multiobjective problem to the problem of solving a system of polynomial equations. For this transformation we use different approaches: one of them without using any optimality conditions, some other using scalarization results and optimality conditions and a last one using multiobjective optimality conditions. We apply the more general optimality conditions that we found to give a methodology for any polynomial problem. However, for specific problems these methodologies allow us to slightly modify the algorithm to incorporate new conditions that could give better result for the problem. This is the case of convex problems.

Other improvement that can be done in this approach is the one of choosing the ordering to compute the Gröbner basis. It is well-known that lexicographical Gröbner bases are the more expensive (in time consuming) to compute. Other elimination orderings could be chosen to improve the computation times. In the case when we apply optimality conditions based on the Lagrangean theory, in many cases, the multipliers associated to each solutions are no unique, so the solutions of the Gröbner system depend on some of these parameters. Comprehensive Gröbner bases may be a good alternative to discuss about the multipliers. As far as we know, there are no implementations of these bases that allow solving problems involving more than 10 variables (decision variables + parameters).

CHAPTER 7

Conclusiones

En esta tesis hemos presentado diversos métodos para resolver dos tipos de problemas multiobjetivo discretos: programas enteros lineales y polinómicos. El caso lineal ha sido abordado haciendo uso de una nueva estructura propuesta también en esta memoria: las bases de Gröbner parciales (en su forma polinomial y en su interpretación geométrica) y de las funciones generatrices. Las bases de Gröbner parciales nos permiten desarrollar una metodología análoga a la presentada previamente por Conti y Traverso pero para problemas multiobjetivo. La ventaja de esta herramienta es que tiene una interpretación geométrica intuitiva, además de relativamente fácil de describir e implementar para obtener las soluciones Pareto-óptimas de un problema multiobjetivo. Por otra parte, la metodología basada en funciones generatrices usa los resultados de Barvinok para codificar los puntos enteros que hay dentro de un politopo para dar una presentación, en términos de funciones racionales cortas, de las soluciones Pareto-óptimas de un problema multiobjetivo. Con esta herramienta, presentamos algunos resultados de complejidad de la optimización entera multiobjetivo, dando un algoritmo polinomial en dimensión fija (sin fijar la dimensión del espacio de objetivos). Esta clase de resultados parecen difíciles de obtener usando bases de Gröbner parciales, ya que incluso en el caso mono-objetvo no está muy clara la complejidad del cálculo de las bases de Gröbner standard (la cota superior conocida es exponencial en la dimensión del espacio de decisión). Sin embargo, la implementación del mejor algoritmo que usa funciones racionales cortas no es fácil de implementar. Para ello, necesitaríamos implementar previamente las operaciones de intersección y proyección con funciones generatrices. Esta tarea es complicada ya que aunque Barvinok da unas forma constructiva de realizar estas operaciones, se utilizan herramientas complejas de geometría discreta. Para las bases de Gröbner parciales, aunque hay algunas subrutinas, el algoritmo para obtener las soluciones Pareto-óptimas no es complicado. Además, la interpretación geométrica de estas bases hace intuitiva y clara la implementación. Otra ventaja de las bases de Gröbner parciales es que hemos detectado algunos patrones para algunos problemas de optimización combinatoria conocidos. Si la estructura de la base

es dada a priori, obtener las soluciones Pareto-óptimas debiera ser bastante rápido, tal y como indican las pruebas computacionales que realizamos. En la práctica, el cálculo de las bases de Gröner parciales consume alrededor del 90% del tiempo total. Además. la interpretación geométrica de las bases de Gröbner parciales de un problema combinatorio coincide con los movimientos en el grafo correspondiente, que permiten realizar mejoras a partir de una solución factible dada. Aún no hemos detectado patrones en las funciones generatrices de ningún problema, ya que sólo algunos problemas pequeños pueden ser calculados a mano con esta herramienta.

La otra clase de problemas que resolvemos son los problemas multiobjetivo polinómicos discretos. Para resolverlos, usamos las propiedades de las bases de Gröbner lexicográficas para resolver sistemas de ecuaciones polinómicas. Para usar estas bases, primero transformamos el problema de optimización en el problema de resolver un sistema de ecuaciones polinómicas. Para esta transformación, usamos diferentes estrategias: una de ellas sin usar condiciones de optimalidad, otras utilizando resultados de escalarización y condiciones de optimalidad para problemas escalares, y finalmente, usando condiciones de no dominancia. Para el desarrollo de estos algorimos hemos considerado las condiciones más generales que hemos encontrado. Sin embargo, para problemas concretos, estas metodologías son mejorables utilizando resultados más fuertes una vez fijadas las propiedades del problema. Nuestros algoritmos son fácilmente adaptables a la incorporación de estas mejoras.

Otra mejora que podría ayudar a estas metodologías es la elección del orden para calcular la bases de Gröbner. Es bien conocido que las bases de Gröbner lexicográficas son las más costosas de calcular. Otros ordenes de eliminación podrían ser elegidos para mejorar los tiempos de computación. En el caso de los algoritmos basados en la Teoría Lagrangiana, en muchos caso, los multiplicadores asociados a cada solución no son únicos, así que las soluciones en estos casos podrían depender de un parámetro. Las bases de Gröbner paramétricas podrían ser una buena alternativa para discutir sobre los multiplicadores. Sin embargo, no parece haber implementaciones que permitan resolver problemas con más de 10 variables (variables + parámetros).

List of Figures

List of Tables

Bibliography

[1] K. Aardal, R. Weismantel, and L.A. Wolsey, *Non-standard approaches to integer programming*, Discrete Appl. Math., 123 (2002), pp. 5–74.

[2] W. Adams and P. Loustaunau, *An Introduction to Gröbner Bases*, American Mathematical Society, 1994.

[3] R. Apéry, *Sur les branches superlinéaires des courbes algébriques*, 1964.

[4] H. Arimura and T. Uno, *A polynomial space and polynomial delay algorithm for enumeration of maximal motifs in a sequence*, in ISSAC'05, 2005, pp. 724–737.

[5] R.M. Baer and O. Østerby, *Algorithms over partially ordered sets*, Journal BIT Numerical Mathematics, 9 (1969), pp. 97–118.

[6] V. Barucci, D.E. Dobbs, and M. Fontana, *Maximality properties in numerical semigroups and applications to one-dimensional analytically irreducible local domains*, 1997.

[7] A. Barvinok, *A polynomial time algorithm for counting integral points in polyhedra when the dimension is fixed*, Mathematics of Operations Research, 19 (1994), pp. 769–779.

[8] A. Barvinok and J.E. Pommersheim, *An algorithmic theory of lattice points in polyhedra*, in New Perspectives in Algebraic Combinatorics, no. 38 in Mathematical Sciences Research Institute Publications, Cambridge Univ. Press, 1999, pp. 91–147.

[9] A. Barvinok and K. Woods, *Short rational generating functions for lattice point problems*, Journal of the American Mathematical Society, 16 (2003), pp. 957–979.

[10] M.S Bazaraa, H.D. Sherali, and C.M. Shetty, *Nonlinear programming : theory and algorithms*, John Wiley and Sons, New York, NY, 1993.

[11] JE Beasley, N Meade, and TJ Chang, *An evolutionary heuristic for the index tracking problem*, European Journal of Operational Research, 148 (1995), pp. 621–643.

[12] A. Beck and M. Teboulle, *Global optimality conditions for quadratic optimization problems with binary constraints*, SIAM J. on Optimization, 11 (2000), pp. 179–188.

[13] T. Becker and V. Weispfenning, *Gröbner bases: a computational approach to commutative algebra*, Springer-Verlag, London, UK, 1993.

[14] D. Bertsimas, D. Perakis, and S. Tayur, *A new algebraic geometry algorithm*

for integer programming, Management Sience, 46 (2000), pp. 999–1008.

[15] D. Bertsimas and R. Weismantel, *Optimization over integers*, Dynamic Ideas, Belmont, Massachusetts, US, 1993.

[16] V. Blanco and J. Puerto, *Short rational functions for multiobjective linear integer programming.* Arxiv, arXiv.[math.OC].0712.4295, December 2007.

[17] V. Blanco and J. Puerto, *A gröbner bases methodology for solving multiobjective polynomial integer programs.* Arxiv, arXiv.[math.OC].0902.1304, December 2008.

[18] V. Blanco and J. Puerto, *Computing the number of numerical semigroups using generating functions.* Arxiv, arXiv.[math.CO].0901.1228, January 2009.

[19] V. Blanco and J. Puerto, *Partial gröbner bases for multiobjective integer linear optimization*, SIAM J. Discrete Math, 3 (2009), pp. 571–595.

[20] M. Bras-Amorós, *Fibonacci-like behavior of the number of numerical semigroups of a given genus*, Semigroup Forum, 76 (2008), pp. 379–384.

[21] K.M. Bretthauer and B. Shetty, *The nonlinear resource allocation problem*, Operations Research, 43 (1995), pp. 670–683.

[22] M. Brion, *Points entiers dans les polyèdres convexes*, Annales scientifiques de l'École Normale Supèrieure Sér 4, 21 (1988), pp. 653–663.

[23] B. Buchberger, *An Algorithm for Finding a Basis for the Residue Class Ring of a Zero-Dimensional Polynomial Ideal*, PhD thesis, University of Innsbruck, Institute for Mathematics, 1965.

[24] B. Buchberger, *Introduction to gröbner bases*, in Gröbner Bases and Applications, B. Buchberger and F. Winkler, eds., vol. 251, London Mathematical Society Lecture Note Series, Cambridge University Press, Cambridge UK, 1997, pp. 32–60.

[25] G. Cantor, *Beiträge zur begründung der transfiniten mengenlehre*, Math. Ann., 49 (1897), pp. 207–246.

[26] A. Cayley, *A theorem on trees*, Quarterly Journal of Mathematics, 23 (1889), pp. 376–378.

[27] YJ Chang and BW Wah, *Polynomial programming using groebner bases*, Int'l Conference on Computer Software and Applications, (1994), pp. 236–241.

[28] V. Chankong and Y.Y. Haimes, *Multiobjective Decision Making Theory and Methodology*, Elsevier Science, New York, 1983.

[29] H. Cohen, *A course in computational algebraic NUMBER theory*, Springer-Verlag New York, Inc., New York, NY, USA, 1993.

[30] P. Conti and C. Traverso, *Buchberger algorithm and integer programming*, in AAECC, 1991, pp. 130–139.

[31] M.W. Cooper, *A survey of methods for pure nonlinear integer programming*, Management Science, 27 (1981), pp. 353–361.

[32] D. Cox, J. Little, and D. O'Shea, *Using Algebraic Geometry*, Springer, 2nd ed., 2005.

[33] David A. Cox, John Little, and Donal O'Shea, *Ideals, Varieties, and Algorithms: An Introduction to Computational Algebraic Geometry and Commutative Algebra, 3/e (Undergraduate Texts in Mathematics)*, Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2007.

[34] H.G. Daellenbach and C.A. De Kluyver, *Note on multiple objective dynamic programming*, Journal of the Operational Research Society, 31 (1980), pp. 591–594.

[35] J.A De Loera, D. Haws, R. Hemmecke, P. Huggins, B. Sturmfels, and R. Yoshida, *Short rational functions for toric algebra and applications*, Journal of Symbolic Computation, 38 (2004), pp. 959–973.

[36] J.A. De Loera, D. Haws, R. Hemmecke, P. Huggins, J. Tauzer, and R. Yoshida, *A user's guide for latte v1.1.* software package LattE is available at http://www.math.ucdavis.edu/latte/, 2003.

[37] J.A. De Loera, R. Hemmecke, and M. Köppe, *Pareto optima of multicriteria integer linear programs.* To appear in: INFORMS Journal on Computing, 2008.

[38] X. Delorme, X. Gandibleux, and F. Degoutin, *Resolution approché du probleme de set packing bi-objectifs*, in Proceedings de l'ecole d'Automne de Recherche Operationnelle de Tours (EARO), 2003, pp. 74–80.

[39] T. Dubé, B. Mishra, and C. Yap, *Complexity of Buchberger's algorithm for Grobner bases*, 1995.

[40] F.Y. Edgeworth, *Mathematical Psychiscs*, P. Keagan, London, 1881.

[41] M. Ehrgott, *Approximation algorithms for combinatorial multicriteria optimization problems*, International Transactions in Operational Research, 7 (2000), pp. 5–31.

[42] M. Ehrgott, J. Figueira, and X. Gandibleux, eds., *Multiobjective Discrete and Combinatorial Optimization*, vol. 147, Annals of Operations Research, 2006.

[43] M. Ehrgott, J. Figueira, and S. Greco, eds., *Multiple Criteria Decision Analysis. State of the Art Surveys*, New York, Springer, 2005.

[44] M. Ehrgott and X. Gandibleux, *A survey and annotated bibliography of multiobjective combinatorial optimization*, OR Spektrum, 22 (2000), pp. 425–460.

[45] M. Ehrgott and X. Gandibleux, eds., *Multiple Criteria Optimization. State of the Art Annotated Bibliographic Surveys*, Boston, Kluwer, 2002.

[46] M. Ehrgott and X. Gandibleux, *Approximative solution methods for multiobjective combinatorial optimization*, TOP, 12 (2004), pp. 1–88.

[47] M. Ehrgott and X. Gandibleux, *Bound sets for biobjective combinatorial optimization problems*, Comput. Oper. Res., 34 (2007), pp. 2674–2694.

[48] M. Ehrgott and D.M. Ryan, *Constructing robust crew schedules with bicriteria optimization*, Journal of Multi-Criteria Decision Analysis, 11 (2002), pp. 139–150.

[49] N. El-Sherbeny, *Resolution of a Vehicle Routing Problem with Multiobjective Simulated Annealing Method*, PhD thesis, Faculte Polytechnique de Mons,

Belgium, 2001.

[50] J.C. Faugère, *A new efficient algorithm for computing gröbner basis (f4)*, Journal of Pure and Applied Algebra, 139 (1999), pp. 61–88.

[51] J.C. Faugère, *A new efficient algorithm for computing gröbner bases without reduction to zero (f5)*, in ISSAC '02: Proceedings of the 2002 international symposium on Symbolic and algebraic computation, New York, NY, USA, 2002, ACM, pp. 75–83.

[52] E. Fernández and J. Puerto, *The multiobjective solution of the uncapacitated plant location problem*, European Journal of Operational Research, 45 (2007), pp. 509–529.

[53] R. Fröberg, *An Introduction to Gröbner Bases*, John Wiley & Sons, 1998.

[54] X. Gandibleux and A. Jaszkiewicz, *Multi-objective metaheuristics*, Journal of Heuristics, 6 (2000), pp. 291–343.

[55] M. R. Garey and D. S. Johnson, *Computers and Intractability: a Guide to the Theory of NP-Completeness*, W. H. Freeman & Co., 1979.

[56] K. Hägglöf, P. Lindberg, and L. Svensson, *Computing global minima to polynomial optimization problems using gröbner bases*, Journal of Global Optimization (Historical Archive), 7 (1995), pp. 115–125.

[57] H. Hamacher and G. Ruhe, *On spanning tree problems with multiple objectives*, Annals of Operations Research, 52 (1994), pp. 209–230.

[58] F. Hausdorff, *Untersuchungen über ordungtypen*, Berichte über die Verhandlungen der königlich sächsischen Gesellschaft der Wissenschaften zu Leipzig, Matematisch - Physische Klasse, 58 (1906), pp. 106–169.

[59] S. Hosten, *Degrees of Gröbner bases of integer programs*, PhD thesis, Cornell University, 1997.

[60] S. Hosten and B. Sturmfels, *GRIN: An implementatiion of grobner bases for integer programming*, in IPCO: 4th Integer Programming and Combinatorial Optimization Conference, 1995.

[61] S. Hosten and R. Thomas, *Gröbner bases in integer programming*, 1998.

[62] H. Ishibuchi and T. Murata, *A multi-objective genetic local search algorithm and its application to flowshop scheduling*, IEEE Trans. Syst., Man, Cybern. C, 28 (1998), pp. 392–403.

[63] J. Jahn, *Vector Optimization: Theory, Applications and Extensions*, Springer, 2004.

[64] N.J. Jobst, M.D. Horniman, C.A. Lucas, and G. Mitra, *Computational aspects of alternative portfolio selection models in the presence of discrete asset choice constraints*, Quantitative Finance, 1 (2001), pp. 489–501.

[65] D. S. Johnson and C. H. Papadimitriou, *On generating all maximal independent sets*, Inf. Process. Lett., 27 (1988), pp. 119–123.

[66] N. Jozefowiez, F. Semet, and E-G. Talbi, *A multi-objective evolutionary algorithm for the covering tour problem*, in Applications of multi-objective evolutionary algorithms, C.A. Coello and G.B Lamont, eds., World Scientific, 2004, pp. 247–267.

[67] O. V. Khamisov, *Algebraic solution of the problems of nonconvex quadratic programming*, Autom. Remote Control, 65 (2004), pp. 218–226.

[68] A.G. Khovanskii and A.V. Pukhlikov, *The riemann-roch theorem for integrals and sums of quasipolynomials on virtual polytopes*, Translation in St. Petersburg Mathematical Journal, 4 (1992), pp. 188–216.

[69] J. Krarup and P.M. Pruzan, *The simple plant location problem: survey and synthesis*, European Journal of Operational Research, 12 (1983), pp. 36–81.

[70] E. Kunz, *über dir klassifikation numerischer halbgruppen*, Regensburger matematische schriften, 11 (1987).

[71] J.B. Lasserre, *Integer programming, barvinok's counting algorithm and gomory relaxations*, Operations Research Letters, 32 (2003), pp. 133–137.

[72] DJ Laughhunn, *Quadratic binary programming with application to capital-budgeting problems*, Operations Research, 18 (1970), pp. 454–461.

[73] J. Lawrence, *Discrete and Computational Geometry*, Discrete Mathematics and Theoretical Computer Science, 6, American Mathematical Society, Providence, RI, 1991, ch. Rational-function-valued valuations on polyhedra, pp. 199–208.

[74] A. K. Lenstra, H. W. Lenstra, and L. Lovàsz, *Factoring polynomials with rational coefficients*, Math. Ann., 261 (1982), pp. 515–534.

[75] H.W. (Jr.) Lenstra, *Integer programming with a fixed number of variables*, Tech. Report Report 81–03, Mathematisch Instituut, Universiteit ban Amsterdam, 1981.

[76] D. Li and X. Sun, *Nonlinear Integer Programming*, International Series in Operations Research & Management Science, Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[77] L. Lovàsz, *An algorithmic theory of NUMBERs, graphs and convexity*, vol. 50, SIAM lecture series, 1986.

[78] D. Maclagan, R.R Thomas., T. Puthenpurackel, A.V. Jayanthan, A. Khetan, L. Gold, and S. Faridi, *Lecture notes for the graduate school at the international conference on commutative algebra & combinatorics h.r.i., allahabad, december 8-13*, 2003.

[79] O. Marcotte and R.M. Soland, *An interactive branch-and-bound algorithm for multiple criteria optimization*, Management Science, 32 (1986), pp. 61–75.

[80] G. Mavrotas and D. Diakoulaki, *A branch and bound algorithm for mixed zero-one multiple objective linear programming*, European Journal of Operational Research, 107 (1998), pp. 530–541.

[81] K. Miettinen, *Nonlinear Multiobjective Optimization*, Kluwer Academic Publishers, Boston, 1999.

[82] GL Nemhauser and LA Wolsey, *Integer Programming and Combinatorial Optimization*, Wiley, New York, 1988.

[83] Jiawang Nie, James W. Demmel, and Victoria Powers, *Minimizing polynomials over semialgebraic sets*, 2005.

[84] A. Ovchinnikov and A. Zobnin, *Classification and applications of monomial*

orderings and the properties of differential term-ordering, in Proceedings of CASC, 2002.

[85] V. Pareto, *Manuel d'Economie Politique*, Marcel Giard, Paris, 1909.

[86] L. Pottier, *Minimal solutions of linear diophantine systems: Bounds and algorithms*, in Proceedings 4th Conference on Rewriting Techniques and Applications, Como (Italy), R. V. Book, ed., vol. 488, Springer-Verlag, 1991, pp. 162–173.

[87] A. Przybylski, X. Gandibleux, and M. Ehrgott, *Two phase algorithms for the biobjective assignment problem*, European Journal of Operational Research, 185 (2008), pp. 509–533.

[88] J.C. Rosales and P.A. García-Sánchez, *Finitely Generated Commutative Monoids*, Nova Science Publishers, New York, NY, 1999.

[89] J.C. Rosales and P.A. García-Sánchez, *Numerical semigroups*, Springer, New York, NY, 2009.

[90] J.C. Rosales, P.A. García-Sánchez, García-García, J.I., and M.B. Branco, *Systems of inequalities and numerical semigroups*, J. London Math. Soc., 65 (2002), pp. 611–623.

[91] H. S. Ryoo and N. V. Sahinidis, *Global optimization of nonconvex NLPs and MINLPs with applications in process design*, Computers and Chemical Engineering, 19 (1995), pp. 551–566.

[92] Y. Sawaragi, H. Nakayama, and T. Tanino, *Theory of Multiobjective Optimization*, Academic Press, 1985.

[93] A. Schrijver, *Combinatorial Optimization. Polyhedra and Efficiency*, Springer, New York, NY, 2003.

[94] E.S. Selmer, *On a linear diophantine problem of frobenius*, J. Reine Angew. Math., 293/294 (1977), pp. 1–17.

[95] S. Steiner and T. Radzik, *Computing all efficient solutions of the biobjective minimum spanning tree problem*, Comput. Oper. Res., 35 (2008), pp. 198–211.

[96] J.R. Stembridge, *A Maple package for posets.*, 2006.

[97] R.E. Steuer, *Multiple Criteria Optimization: Theory, Computation and Application*, John Wiley & Sons, New York, NY, 1985.

[98] B. Sturmfels, *Grobner Bases and Convex Polytopes*, no. 8 in Univ. Lectures Series, Providence, Rhode Island, 1996, American Mathematical Society, 1996.

[99] B. Sturmfels, *Solving systems of polynomial equations*, Tech. Report 97, CBMS Regional Conference Series of the AMS, Rhode Island, 2002.

[100] B. Sturmfels, *Algebraic recipes for integer programming*, in Proceedings of Symposia in Applied Mathematics, vol. 61, 2004.

[101] B. Sturmfels and R.R. Thomas, *Variation of cost functions in integer programming*, Mathematical Programming, 77 (1997), pp. 357–387.

[102] M. Tawarmalani and N.V. Sahinidis, *Global optimization of mixed integer nonlinear programs: a theoretical and computational study*, 2003.

[103] R.R. Thomas, *A geometric Buchberger algorithm for integer programming*, Mathematics of Operations Research, 20 (1995), pp. 864–884.

[104] R.R. Thomas, *Applications to integer programming*, in PSAM: Proceedings of the 53th Symposium in Applied Mathematics, American Mathematical Society, 1998.

[105] R.R. Thomas and R. Weismantel, *Truncated gröbner bases for integer programming*, Applicable Algebra in Engineering, Communication and Computing, 8 (1997), pp. 241–256.

[106] S. Tsukiyama, M. Ide, H. Ariyoshi, and I. Shirakawa, *A new algorithm for generating all maximal independent sets*, SIAM J. Comput., 6 (1979), pp. 505–517.

[107] E. Ulungu and J. Teghem, *The two-phases method: An efficient procedure to solve biobjective combinatorial optimization problems*, Foundations of Computing and Decision Sciences, 20 (1995), pp. 149–165.

[108] R. Urbaniak, R. Weismantel, and G.M. Ziegler, *A variant of the buchberger algorithm for integer programming*, SIAM J. Discret. Math., 10 (1997), pp. 96–108.

[109] S. Verdoolaegehttp, *Software* ***barvinok***. http://www.kotnet.org/~skimo/barvinok/, 2008.

[110] B. Villarreal and Karwan, *Multicriteria dynamic programming with an application to the integer case*, Journal of Optimization Theory and Applications, 31 (1982), pp. 43–69.

[111] M. Visée, J. Teghem, M. Pirlot, and E. L. Ulungu, *Two-phases method and branch and bound procedures to solve the biobjective knapsack problem*, J. of Global Optimization, 12 (1998), pp. 139–155.

[112] K. Woods and R. Yoshida, *Short rational generating functions and their applications to integer programming*, SIAG/OPT Views and News, 16 (2005), pp. 15–19.

[113] P.L. Yu, *Cone convexity, cone extreme points, and nondominated solutions in decision problems with multiobjectives*, Journal of Optimization Theory and Applications, 14 (1974), pp. 319–377.

[114] G.M. Ziegler, *Gröbner bases and integer programming*, in Some Tapas of Computer Algebra, A.M. Cohen, H. Cuypers, and H. Sterk, eds., Springer-Verlag, Berlin, FRG, 1999, ch. 7, pp. 168–183.

[115] S. Zionts, *A survey of multiple criteria integer programming methods*, Annals of Discrete Mathematics, 5 (1979), pp. 389–398.

[116] S. Zionts and J. Wallenius, *Identifying efficient vectors: some theory and computational results*, Operations Research, 23 (1980), pp. 785–793.


UNIVERSIDAD DE SEVILLA