UNIVERSIDAD DE CONCEPCIÓN
FACULTAD DE INGENIERÍA
DEPARTAMENTO DE INGENIERÍA CIVIL INDUSTRIAL

Una metaheurística para resolver el polynomial robust knapsack problem

Por

Daniel Antonio Castillo Molina

Memoria de Título presentada a la Facultad de Ingeniería de la Universidad de Concepción para optar al título profesional de Ingeniero Civil Industrial

Enero 2023
Concepción, Chile

Profesor Guía:

Carlos Contreras Bolton

Para las personas que estan y las que no.

Agradecimientos

Quiero empezar agradeciendo a la vida, que ha puesto todo en su lugar y cada día aprendo cosas nuevas de ella, gracias por los desafíos, las caídas, los momentos de felicidad, los de calma, los de reflexión, los de estrés, los de pena, los de alegría y los de amor, estas hecha de momentos y agradezco que me enseñes a apreciarlos.

También quiero agradecer a mis padre y mi madre, Miguel y Gloria, porque gracias a su amor y apoyo incondicional he tenido el coraje para experimentar y para lograr los objetivos propuestos. Gracias por siempre estar conmigo ya sea física o espiritualmente, gracias por crear el espacio seguro para mi crecimiento y ser un ejemplo de esfuerzo, de sacrificio y de amor.

Agradecer a mi hermano, Miguel, que es mi mejor amigo y una fuente de inspiración, gracias por tu paciencia, por tu energía, por tus enseñanzas y por la protección, gracias por dejarme ser un hermano pequeño y poder siempre contar con mi hermano mayor, estamos grandes y quizá nos separemos físicamente, pero, siempre te llevo conmigo. Agradecer a Lidia su pareja, mi hermana, desde que llegaste a la familia solo has sumado, momentos, ideas, emociones y ternura, eres un apoyo y una amiga, y estoy feliz de verte a ti y a mi hermano crecer juntos.

Agradecer a mis amigos, los que estan y los que no, a los que la vida nos junta y a los que nos separó, todos han sido parte del camino y los llevaré por siempre en mi corazón, especial mención a Thomas, Santiago y Javiera, que son un pilar en mi vida, gracias por tenerme paciencia, por comprenderme y por retarme cuando ven que estoy equivocado, atesoro su cariño y su honestidad.

Muchas gracias a mi profesor Guía, Carlos Contreras, que siempre ha tenido la disposición y la paciencia para ayudarme, gracias profesor por la comprensión y el apoyo.

Resumen

El polynomial robust knapsack problem (PRKP) es una variante del clásico knapsack problem (KP), que consiste en elegir un subconjunto de elementos que maximice la utilidad, sujetos a restricciones de capacidad. En el contexto del PRKP, los elementos exhiben un beneficio independiente y uno dependiente de otros elementos seleccionados, denominado sinergía. Esto refleja situaciones del mundo real, tales como la construcción de un estacionamiento al lado de una estación de metro concurrida, proyectos que se benefician mutuamente. Además, los costos asociados a la elección de cada elemento son inciertos y varían en un rango determinado. Esta característica confiere al PRKP una complejidad superior en comparación con el KP tradicional. Dada su aplicabilidad en diversos campos, especialmente en inversiones, la investigación de este problema adquiere gran relevancia. Por tanto, esta memoria de título presenta una heurística para abordar la resolución del PRKP. Se propone una metodología que emplea soluciones parciales generadas mediante un muestreo aleatorio, generando instancias de menor envergadura que son posteriormente resueltas mediante el empleo de un solver de proposito general. El algoritmo propuesto es validado en 1600 instancias y se comparan con otros algoritmos del estado del arte. Los resultados indican un rendimiento competitivo, en muchos casos aproximándose a las soluciones obtenidas por los métodos de la literatura. Cabe mencionar que se obtiene una diferencia de valor obtenido para la función objetivo, igual o inferior al 5 % en comparación con el solver de propósito general.

Keywords – polynomial robust knapsack problem, knapsack problem, heurística, algoritmo, soluciones parciales

Abstract

The polynomial robust knapsack problem (PRKP) is a variant of the classic knapsack problem (KP), which involves choosing a subset of elements to maximize utility, subject to capacity constraints. In the context of PRKP, elements exhibit both independent and dependent benefits on other selected elements, known as synergy. This reflects real-world situations, such as building a parking lot next to a busy subway station, where projects mutually benefit each other. Additionally, the costs associated with choosing each element are uncertain and vary within a specified range. This characteristic adds complexity to PRKP compared to the traditional KP. Due to its applicability in various fields, particularly in investments, researching this problem becomes highly relevant. Therefore, this thesis presents a heuristic to address the resolution of PRKP. A methodology is proposed that uses partial solutions generated through random sampling, creating smaller instances that are subsequently solved using a general-purpose solver. The proposed algorithm is validated on 1600 instances and compared with other state-of-the-art algorithms. The results indicate competitive performance, in many cases approaching to the results obtained by the methods of the state-of-the-art. It is worth mentioning that a difference in the objective function value is obtained, equal to or less than 5 %, compared to the general-purpose solver.

Keywords – polynomial robust knapsack problem, knapsack problem, heuristic, algorithm, partial solutions

Índice General

Índice de Tablas

Índice de Algoritmos

Capítulo 1

Introducción

Este capítulo presenta el polynimial robust knapsack problem y la motivación de este trabajo. Además, se presenta el objetivo general y los específicos.

1.1. Motivación

Imaginemos una elección simple en la que tenemos que optar entre dos posibilidades. Por ejemplo, decidir si viajar en automóvil o en bus, elegir entre comprar o arrendar una casa, o decidir si vacacionar dentro del país o en el extranjero. Después de considerar estos ejemplos, es evidente que nuestras vidas están llenas de decisiones similares: decisiones en las que tenemos que elegir entre dos opciones. Estas elecciones son comunes en nuestra vida personal y también frecuentes en el ámbito profesional. Para hacer más sencilla la toma de decisiones de este tipo, simplificamos la elección al identificar dos aspectos clave de cada opción: el beneficio y su costo. A partir de estas consideraciones, se puede construir un modelo matemático de optimización que nos indique la mejor elección entre las disponibles, según los supuestos establecidos.

El problema de la mochila, conocido también como knapsack problem (KP) en inglés, es uno de los problemas clásicos en la programación matemática, empleando variables binarias. A pesar de su simplicidad aparente, su relevancia perdura a lo largo del tiempo. Su formulación clásica involucra una mochila con capacidad limitada y un conjunto de elementos. Cada elemento está caracterizado por un valor y un peso asociado, siendo el objetivo seleccionar un subconjunto de elementos de que maximice el valor total sin sobrepasar la capacidad de la mochila.

El KP y sus extensiones siguen siendo objeto de investigación, no solo por su utilidad en la resolución de problemas, sino que también por su aplicabilidad en contextos sociales

y por su rol como subproblemas en la solución de desafíos de mayor complejidad. Entre las variantes más destacadas y conocidas se incluyen el subset sum problem, el unbounded KP, el Multiple KP y el multidimensional KP, discutidos en Kellerer & Strusevich (2008).

Recientemente, Baldo *et al.* (2023) presentaron el polynomial robust KP (PRKP), que consiste en una extensión que aborda dos aspectos cruciales del KP: la robustez de los costos y las sinergias polinomiales entre los ítems. Los autores proponen dos métodos para resolver el PRKP, una heurística basada en machine learning y un algoritmo genético. El PRKP abre nuevas posibilidades y enfoques para abordar la incertidumbre y las sinergias en contextos de decisión complejos.

El PRKP se basa en el robust knapsack problem (RKP) propuesto por Eilon (1987), quien demostró cómo una solución óptima del KP podría volverse inviable cuando se introduce la incertidumbre en el presupuesto total. Por lo que, se introdujo un rango de valores para el presupuesto y se propuso un indicador de estabilidad para guiar la toma de decisiones. Generando soluciones robustas que son, en cierto grado, inmunes a la incertidumbre. Además, desarrolló un diagrama de portafolio para ordenar los proyectos en función de su relevancia. Por otro lado, otra variante que se relaciona con el PRKP es el quadratic KP (QKP), que fue propuesta por Gallo *et al.* (1980). El QKP considera a dos o más alternativas que producen beneficios en conjunto (sinergia). Esto genera que la función objetivo se convierta en una función cuadrática o de mayor grado, que se puede simplificar mediante la linealización Lagrangiana, generando así, el PKP. Los autores introdujeron la aplicación de planos superiores en el método de branch & bound para abordar el QKP.

El PRKP tiene múltiples aplicaciones en inversión de proyectos (Baldo *et al.*, 2023). Un ejemplo de aplicación, cuando los tomadores de decisiones públicas se enfrentan al desafío de seleccionar un subconjunto de proyectos provenientes de un amplio abanico de opciones, con el objetivo de mejorar las condiciones de vida de la población. En este contexto, sus elecciones deben ser robustas ante la incertidumbre, garantizando que incluso en situaciones desfavorables, los proyectos sigan siendo viables dentro del presupuesto establecido. Además, los tomadores de decisiones son conscientes de que diversos proyectos pueden generar efectos sinérgicos, tanto positivos como negativos. Por ejemplo, instalar estaciones de carga eléctrica en una zona de estacionamiento y la rehabilitación de un edificio cercano pueden ofrecer a la comunidad un beneficio mayor que la simple suma de ambos proyectos, destacando la importancia de considerar estas interacciones al tomar decisiones estratégicas.

Otro ejemplo de aplicación, es cuando un tomador de decisiones se enfrenta al desafío de realizar inversiones estratégicas en un subconjunto de macros sectores. Simultáneamente,

debe disponer de estadísticas relevantes sobre estos sectores, como costos de inversión, rendimientos, volatilidades, entre otros. Este modelo considera la utilidad esperada futura, el costo nominal actual del activo, un límite superior para el costo del activo, y debe tener en cuenta su volatilidad histórica. Además, pueden surgir sinergias de las correlaciones entre activos, favoreciendo la diversificación y la reducción del riesgo en la toma de decisiones de inversión.

En el primer ejemplo, la solución del modelo considera el conjunto de proyectos que maximice el beneficio y los costos variables de los proyectos, asegurando que la decisión se ajuste al presupuesto disponible y mejore significativamente la calidad de vida de la población. Mientras, en el segundo ejemplo, la solución considera el mejor portafolio de inversiones, tomando en cuenta el riesgo presente, la proyección futura de este, ajustándose a la restricción de presupuesto y maximizando la utilidad obtenida.

El PRKP pertenece a la familia de problemas NP-hard, debido a esto el PRKP es computacionalmente complejo. Por tanto, es importante diseñar nuevos y mejores algoritmos para encontrar mejores soluciones al problema, esto podría ayudar a industrias constructoras, instituciones financieras o el gobierno, que tuvieran el problema específico. Además, puede permitir el desarrollo de nuevas estrategias para resolver problemas similares pertenecientes a la familia NP-hard.

En el marco de esta investigación, se propone una metaheurística para abordar eficazmente el PRKP. Esta metaheurística opera buscando soluciones parciales para subconjuntos del problema, identificando y seleccionando las variables más prometedoras. Estas soluciones parciales se presentan como entradas para un método de resolución. Para evaluar y validar la eficacia de la metaheurística propuesta, se llevan a cabo experimentos computacionales exhaustivos utilizando instancias del problema que varían en tamaño, abarcando desde 300 hasta 1500 elementos.

1.2. Objetivo general

Implementar una metaheurística para el PRKP.

1.3. Objetivos específicos

- Revisar el estado del arte de los métodos para resolver el PRKP.
- Diseñar una metaheurística para el PRKP.

- Implementar la metaheurística diseñada.
- Medir el desempeño del método propuesto.
- Comparar los resultados obtenidos con el estado del arte.

1.4. Estructura del documento

El resto del documento se estructura de la siguiente forma. En el capítulo siguiente se presenta formalmente el problema, incluyendo modelos matemáticos relacionados y una revisión de la literatura de métodos que abordan el PRKP y similares. En el Capítulo 3, se presenta la metaheurística diseñada, sus componentes y explicación. El Capítulo 4 aborda los experimentos computacionales y resultados obtenidos, que luego es discutido en el Capítulo 5.

Capítulo 2

Polynomial robust knapsack problem

Este capítulo presenta formalmente el problema y sus modelos matemáticos. Además, se revisa la literatura asociada al PRKP y problemas similares, con enfoque en los algoritmos del estado del arte que resuelven a estos.

2.1. Descripción del problema

Formalmente, el PRKP es definido con un conjunto de n ítems denotado por $I = \{0, 1, 2, 3, \ldots, n\}$. Cada ítem i tiene un beneficio (profit) ($p_i \in \mathbb{R}$) y un valor c_i que oscila dentro del intervalo $[c_i^{min}, c_i^{max}]$. Donde c_i^{min} es costo nominal y c_i^{max} es costo variable. Además, existe un conjunto de ganancia que entrega cada sinergia denominado $A = \{\overline{I}_j \backslash I_j \subseteq I : j \in 2^{|I|}\}$, donde la sinergia es el beneficio que entrega uno o más elementos al elegirse de manera simultánea. Cada sinergia $a \in A$ tiene un beneficio asociado B_a. Finalmente, W es la capacidad o el presupuesto disponible.

El objetivo es encontrar un subconjunto ítems $S \subseteq I$ y un subconjunto de sinergias que maximice la utilidad, respetando el presupuesto. Donde la utilidad es la suma de los beneficios de S más la suma de los beneficios de las sinergias en S, restando el costo del peor caso de S considerando cualquier variación estocástica. Para S considerarse solución factible, el costo del peor caso de S debe ser menor a la capacidad W.

2.2. Modelo de Programación Matemática

El PRKP puede ser formulado mediante un modelo de programación lineal entera mixta y fue propuesto por Baldo *et al.* (2023). Así, en la Tabla 1 se presenta la nomenclatura

del modelo para el PRKP, con sus conjuntos, parámetros y variables de decisión. A continuación, se describe el modelo.

Tabla 1: Nomenclatura usada en la definición del modelo existente.

Conjuntos	
I	Conjunto de ítems disponibles $\{0, 1, 2, 3, \ldots, n\}$.
A	Conjunto de las sinergias entre ítems, $A \subset 2^{\lvert I \rvert}$.
Parámetros	
c_i	Costo individual del ítem $i \in I$, $c_i \in [c_i^{min}, c_i^{max}]$.
p_i	Beneficio individual entregado por cada ítem $i \in I$.
B_a	Beneficio de la sinergia entre ítems $a \in A$.
Γ	Cantidad máxima de ítems que cambian su valor nominal por el valor variable.
Variables de decisión	
x_i	Toma valor 1, si el ítem $i \in I$ es seleccionado, 0 caso contrario.
z_a	Toma valor 1, si la sinergia $a \in A$: $(i \in a) \in I$ y pertenece a la solución, 0 caso contrario.
ρ	Variable dual asociada a las restricciones generadas al tomar el peor escenario.
π_i	Variable dual asociada a la variación de costo para cada elemento $i \in I$.

En el modelo propuesto, se introducen diversas variables y conjuntos para representar las características y restricciones del problema de optimización. En primer lugar, los conjuntos I y A representan los ítem disponibles y las sinergias entre ellos, respectivamente. Los parámetros c_i, p_i y B_a denotan el costo individual, el beneficio individual y el beneficio de las sinergias, respectivamente, los cuales son definidos de antemano por la instancia del problema. Las variables de decisión x_i indican si un ítem i es seleccionado o no, mientras que z_a es una variable auxiliar introducida para linealizar el problema, representando la activación de las sinergias. Específicamente, cada variable z_a representa la multiplicación de múltiples variables x_i que participan en la sinergia a. Además, ρ y π_i son variables duales asociadas a las restricciones generadas al considerar el peor escenario en la variación de costos para cada elemento.

$$\text{maximizar} \quad \sum_{i \in I} p_i x_i + \sum_{a \in A} B_a z_a - \sum_{i \in I} c_i^{min} x_i - (\Gamma \rho + \sum_{i \in I} \pi_i) \tag{1}$$

sujeto a:

$$\sum_{i \in I} c_i^{min} x_i + \Gamma \rho + \sum_{i \in I} \pi_i \leq W \tag{2}$$

$$\rho + \pi_i \geq (c_i^{max} - c_i^{min}) x_i \quad \forall i \in I \tag{3}$$

$$\sum_{i \in a} x_i \leq |a| - 1 + z_a \quad \forall a \in A : B_a < 0 \tag{4}$$

$$z_a \leq x_i \quad \forall i \ , \forall a \in A : B_a \geq 0 \tag{5}$$

$$x_i \in \{0, 1\} \quad \forall i \in I \tag{6}$$

$$z_a \in \{0, 1\} \quad \forall a \in A \subseteq 2^{|I|} \tag{7}$$

$$\pi_i \geq 0 \quad \forall i \in I \tag{8}$$

$$\rho \geq 0 \tag{9}$$

La función objetivo (1) representa la utilidad, los términos positivos son el beneficio individual de cada ítem elegido y el beneficio de las sinergias que participan. Los términos negativos representan el costo total, donde el primer término es el costo fijo de cada ítem elegido (límite inferior del intervalo costo) y el segundo término es el costo variable del ítem, que representa el problema dual resultante de aplicar la transformación dual al problema de minimizar los costos variables. La restricción de tipo (2) corresponde a la capacidad de la mochila que es la suma de los costos tiene que ser menor al presupuesto. Las restricciones de tipo (3) provienen del problema dual que representa el problema de los costos variables y genera que el problema se evalúe en el peor escenario. Esta evaluación en el peor caso busca generar soluciones robustas, ya que si una solución es efectiva en condiciones extremas, es probable que también funcione bien en situaciones más favorables. Por tanto, cuando existe la máxima variabilidad en los costos para a lo más Γ términos. Estas restricciones consideran el mayor costo para cada x_i elegido. Las restricciones de tipo (4) y (5) son la activación de las sinergias. Donde las restricciones (4) imponen que la única forma de elegir una sinergia con beneficio negativo es cuando todos los elementos participantes de esta sinergia son elegidos como parte de la solución final. Las restricciones (5) corresponden a la sinergia positiva que es elegida cuando todos los ítems participantes son elegidos en la solución final. Las restricciones (6)-(9) corresponden a los dominios de las variables.

2.3. Revisión bibliográfica

El PRKP es planteado por primera vez en Baldo *et al.* (2023) y no existe más literatura. Por tanto, se revisan versiones del KP que influyen en la formulación y resolución del principal problema estudiado en el informe como el QKP, PKP y el RKP.

Baldo *et al.* (2023) presentan el PRKP y su modelo de programación lineal entera mixta que consiste en la linealización del QKP y la característica robusta del RKP, con énfasis en las dos características importantes, la robustes y las sinergias. Baldo *et al.* (2023) proponen dos heurísticas, una heurística basada en machine learning (HML) y un algoritmo genético (AG). La HML propone un clasificador binario que asigna a cada variable la probabilidad de tomar valor 0 o 1. Las variables cuya probabilidad de ser elegida en la solución final es mayor se fijan en el valor 1, mientras, las menos probables son procesadas nuevamente por un solver de propósito general. Así, el solver es beneficiado por la reducción del espacio de solución. El algoritmo considera características como beneficio, costo y sinergias para clasificar variables.

En tanto, el AG genera una población de cromosomas, donde cada cromosoma representa un conjunto único de atributos asociados al problema. La población evoluciona a través de iteraciones que involucran procedimientos de selección de padres, cruce y mutación. La selección se basa en el aporte de las variables (x_i) a la función objetivo del problema. Aunque el AG funciona bien en la mayoría de los casos, se introduce un hiperparámetro de penalización para abordar casos excepcionales.

En Mirshekarian & Šormaz (2018) examinan el uso del machine learning para potenciar algoritmos heurísticos y metaheurísticos en problemas de optimización combinatoria, como el caso del KP. Rezoug *et al.* (2021) usa machine learning para resolver el MKP, predicen si la variable será seleccionada en la solución final. Comparan el rendimiento de un modelo de aprendizaje supervisado y un solver de proposito general aplicados. Ellos proponen aplicar AG a la solución encontrada por el modelo de machine learning, para encontrar mejores resultados, destacando el complemento de estos dos métodos.

El RKP es un problema ampliamente estudiado y fue introducido por primera vez en Eilon (1987). Donde se muestra como la solución del KP puede variar de manera significativa ante un pequeño cambio en el presupuesto o los costos. Dado que uno de los puntos ciegos de la programación lineal es que los problemas son determinísticos, es decir, se sabe con 100 % de certeza los parámetros de entrada, un escenario difícil de encontrar en situaciones reales. Por tanto, la incertidumbre es una de las características principales del PRKP y el RKP y se presenta en el valor de los costos, ingresos o

presupuesto. Esto es común en varios problemas de programación lineal, como en Ben-Tal & Nemirovski (2000), que estudian múltiples problemas de optimización que son afectados por la incertidumbre, cuando existe incertidumbre en los datos, como el costo, los resultados pierden su optimalidad, convirtiéndose incluso en resultados infactibles. Por ello, aplican el método de solución robusta (inmune a la incertidumbre) en 90 problemas de programación lineal y demuestran que el método produce una solución subóptima e inmune (hasta cierto punto) a la incertidumbre.

Una de las metodologías expuestas en Ben-Tal & Nemirovski (2000) es resolver los problemas en el peor escenario, cuando los parámetros varían dentro de un intervalo. En el PRKP corresponden a los costos que pertenecen a un intervalo, tomando el costo superior de éstos como el peor escenario para ser resuelto. Los autores proponen un nuevo estimador lineal para estimar un vector de parámetros desconocido en un modelo lineal con incertidumbres de datos acotadas. El estimador está diseñado para minimizar la diferencia entre el error cuadrático medio y el error cuadrático óptimo, mejorando la calidad de las soluciones propuestas por otros algoritmos.

Nobibon & Leus (2013) estudian el RKP y usan una perspectiva diferente, evaluando el problema cuando el beneficio o el costo son variables. Al contrario de otros trabajos que evaluaban como variable solo el presupuesto o los costos. Proponen algoritmos de relajación para resolver el problema, cuando este está restringido superiormente, o sea, hay una cantidad máxima de ítems que pueden variar, y también, cuando no está restringido. Demostrando que el algoritmo de relajación es eficiente a la hora de encontrar una solución óptima en ambos escenarios.

Poss (2018) aborda problemas de optimización combinatoria robusta, tanto de minimización como de maximización, en el contexto de un conjunto de incertidumbre definido por restricciones de mochila. Se introducen algoritmos exactos y de aproximación que extienden las propuestas iterativas de Bertsimas & Sim (2003). Se analizan las limitaciones de éstos algoritmos, identificando situaciones complejas en las cuales no son eficientes. Además, desarrollan un esquema de aproximación para el problema robusto al aproximar elipsoides alineados con los ejes mediante restricciones de mochila.

Büsing *et al.* (2019) fusionan el conjunto de incertidumbres presupuestarias con una estrategia de acción de recuperación, que permite eliminar hasta k elementos para restablecer la viabilidad cuando se conocen los coeficientes reales. Presentan tres enfoques diferentes para formular el recoverable robust KP (rrKP) mediante programación lineal entera, destacando una reformulación compacta de tamaño cuadrático. En un análisis experimental para resolver el rrKP sobre un conjunto seleccionado de instancias de

referencia. Los resultados demuestran que el enfoque propuesto supera a otras alternativas.

En Han *et al.* (2015) expanden la búsqueda de soluciones para el RKP. Proponen un algoritmo de tiempo polinomial basado en la optimización robusta para encontrar soluciones. Cuando es comparado con otros algoritmos, éste muestra más estabilidad en su eficiencia ante diferentes tipos de tamaños de instancia. Monaci *et al.* (2013) proponen un algoritmo de programación dinámica con técnicas para mejorar la eficiencia para el RKP. La comparación computacional demuestra un rendimiento superior del algoritmo en comparación con otros algoritmos exactos utilizados en la optimización robusta.

Gallo *et al.* (1980) proponen el QKP y proponen el uso de planos superiores derivados de la relajación lagrangiana en el branch & bound. Además, en sus experimentos reportan que estos métodos son útiles para facilitar la búsqueda de soluciones.

En Billionnet *et al.* (1999) se aborda el QKP y proponen un método basado en la descomposición lagrangiana para calcular una cota superior. Los experimentos computacionales respaldan la cota, con un error relativo generalmente inferior al 1 %, incluso en instancias de gran tamaño (hasta 500 elementos).

Muchos métodos son propuestos para la resolución del QKP, como en Hammaer & Rader (1997) que consideran dividir el problema en subproblemas más pequeños. Glover *et al.* (2002) comparan la efectividad de dos linealizaciones comunes con la formulación original del QKP y aplican una búsqueda tabú para resolverlo. Logrando encontrar soluciones óptimas en instancias grandes del QKP. Létocart *et al.* (2012) utilizan una relajación y descomposición lagrangianas para la optimización y la aceleración de la búsqueda de la solución. A pesar de lo anterior, existe escasa atención en el modelamiento polinomial del KP.

Kellerer & Strusevich (2008) proponen una solución eficiente para QKP y sus aplicaciones en la planificación de producción, con un esquema de aproximación polinomial. Esto muestra adaptarse con éxito a problemas de programación de una sola máquina. Estos resultados tienen aplicaciones prácticas en la optimización de los tiempos de finalización.

Capítulo 3

Metaheurística propuesta

Este capítulo presenta el algoritmo propuesto, los pseudocódigos de sus partes principales, describiendo de manera detallada los pasos de la metaheurística propuesta (MH).

3.1. Esquema general

La estrategia empleada por MH es acercarse a la solución óptima a través de la construcción de soluciones parciales. Una solución parcial es definida como una solución intermedia que no representa la solución final, sino más bien un paso hacia su resolución. En esta metaheurística de búsqueda iterada, se exploran subconjuntos del conjunto inicial, lo que conduce al descubrimiento de soluciones que cumplen con las restricciones impuestas, aunque no necesariamente sean óptimas en el modelo original. Esto se debe a que pueden surgir cambios, como la adición de ítems o la selección de un conjunto diferente de variables en la solución final.

La premisa de MH es identificar, mediante soluciones parciales a las variables con mayor probabilidad de incluirse en el conjunto final. La información sobre la probabilidad se integra en el modelo y se resuelve mediante el método de soluciones de propósito general. En cada iteración, se resuelve un subproblema (s) y se almacena temporalmente el subconjunto de ítems seleccionados como parte de la solución. Además, se utiliza un conjunto s^* que representa el conjunto de ítems que genera la mejor solución actual. Estos conjuntos permiten ir mejorando el espacio de búsqueda en cada iteración. Adicionalmente, cada elemento $i \in I$ tiene asociado un puntaje α_i, que representa la frecuencia con la que ha sido elegida como parte de la mejor solución (s^*).

MH es presentada en el presente estudio como Algoritmo 1. Las líneas 1-3 inicializan los

conjuntos y un contador de iteraciones. Luego, en la línea 4, se construye el bucle principal del algoritmo que itera sobre las primeras cuatro funciones que construyen subconjuntos. En las líneas 5-10, se genera un subconjunto $\overline{I}$ mediante una función que proporciona subconjuntos diferentes en cada iteración. Luego, el conjunto generado es unido a la mejor solución encontrada. Con este nuevo subconjunto, se formúla un subproblema y se resuelve en un tiempo límite T, el subconjunto seleccionado como solución es almacenado en s y a las variables seleccionadas en ese subconjunto se le suma uno a su α_i asociado. Posteriormente, en las lineas 7-9, si la solución actual es mayor que la mejor solución encontrada, entonces, la solución actual es la mejor solución. Las lineas 12-13 son el último paso, con la mejor solución encontrada hasta el momento (s^*) y el conjunto de puntajes α. Además, se genera la solución semilla s que se entrega al solver de propósito general y se resuelve en un tiempo límite T.

Algoritmo 1: Esquema General de MH

```
1  s* = s = Ī ← ∅
2  α_i ← 0,∀i ∈ I
3  i ← 1
4  while i ≤ 4 do
5  |  Ī ← subproblema_i(I) ∪ s*
6  |  s, α_i ← solve(Ī, T)
7  |  if f(s) > f(s*) then
8  |  |  s* ← s
9  |  end
10 |  i ← i + 1
11 end
12 s ← generar-solución-inicial(α, s*, I)
13 s ← solve(s, T)
14 return s
```

3.2. Subproblemas del algoritmo

La generación de una solución parcial a partir de subproblemas es la base de MH. Para esto, se proponen soluciones a partir de la creación y resolución de subconjuntos, llamados subproblemas. A continuación, se explica cómo se crean los subconjuntos al aplicar las funciones subproblema_1, subproblema_2, subproblema_3 y subproblema_4.

- subproblema_1: se construye este subconjunto basado en la utilidad individual que entrega cada ítem, definida como la ganancia menos el costo (límite superior del

intervalo costo). Los elementos se ordenan de mayor a menor según su utilidad y se selecciona el 70 %. En caso de que sea negativa el ítem se descarta del subconjunto. La utilidad de cada ítem se calcula como $B_i - C_i^{max}, \quad \forall i \in I$.

- subproblema$_2$: este subconjunto considera el beneficio total que generan los ítems, incluyendo el beneficio de las sinergias en las que participa. Este nuevo beneficio, es calculado como $\sum_{i \in a} B_i - \sum_{i \in a} C_i + B_a, \quad \forall a \in A$.

- subproblema$_3$: se crea de manera completamente aleatorio, asignando a cada elemento del conjunto general igual probabilidad de ser elegido.

- subproblema$_4$: se construye utilizando la probabilidad de que una variable sea elegida, se calcula como el puntaje obtenido dividido la suma total de los puntajes. Esto representa el peso que tiene la variable sobre el resto.

Al iterar sobre cada uno de estos subconjuntos, se obtiene el puntaje final de cada variable α_i ($i \in I$) y el mejor subconjunto de ítems (s^*).

3.3. Generación de la solución inicial

La solución inicial parte desde una solución relajada, donde, se relaja el dominio de las variables para que tomen valores continuos entre 0 y 1, en vez de que tomen valores binarios. Luego, se entrega al solver de propósito general como solución inicial, esto es generado en la linea 12 del Algoritmo 1. De esta forma, se reduce el espacio de búsqueda y se acelera la convergencia a una solución mejor.

Esta solución inicial, se crea a partir del puntaje para cada variable α_i y el mejor subconjunto de ítems encontrado s^*. La formación de la solución considera la probabilidad de la variable a ser seleccionada en la solución final, evaluando esta probabilidad mediante un parámetro β_i definido como $\beta_i = \frac{p_i - c_i}{\mathrm{W}} \times \frac{\alpha_i}{\sum \alpha_i}$. El valor de β_i representa el peso de la variable en la solución final, ya que refleja la contribución de utilidad en relación con el presupuesto y el peso de la variable en comparación con el resto de los ítems.

Con lo anterior, los elementos i que cumplen $\{i \backslash \beta_i > 0,5\}$ y son parte del mejor subconjunto encontrado, son seleccionadas como parte de la solución inicial del modelo en el Algoritmo 1.

Capítulo 4

Resultados

Este capítulo presenta los resultados obtenidos de la metaheurística propuesta y se evalúa la eficiencia con respecto a los métodos propuestos en la literatura.

4.1. Instancias de prueba

Las instancias utilizadas corresponden a las instancias propuestas por Baldo *et al.* (2023). Se consideran 1600 instancias, con un tamaño que varía entre 100 y 1500 ítems a elegir. En la Tabla 2 se presenta un resumen con el tamaño de las instancias, cantidad de instancias por tamaño, su clasificación y el presupuesto promedio para cada una.

Tabla 2: Descripción instancias de prueba.

Tamaño	Cantidad de Instancias	Presupuesto promedio
100	200	928,27
300	200	2765,15
500	200	4812,62
700	200	6562,95
900	200	8432,23
1100	200	10057,58
1300	200	11828,50
1500	200	14094,77

4.2. Parámetros del algoritmo y su calibración

Los parámetros involucrados son el tiempo de iteración T en cada función y un valor β_i asociado a cada elemento. Ambos parámetros son calibrados mediante iteraciones de la

metaheurística aplicada a un subconjunto de instancias.

El tiempo de iteración T es el tiempo límite que el solver puede iterar en cada una de las funciones asociadas en la metaheurística. La heurística se ejecutó considerando tres tiempos diferentes: 7, 12 y 17 segundos. Se elige el tiempo con mejores resultados y demostró ser competitivo con los métodos propuestos en la literatura. El tiempo elegido fue de 12 segundos. Mientras, para la selección de β_i, se propusieron tres fórmulas (10), (11) y (12) para calcularlo y se elige el β_i que entrega una mayor cantidad de mejores soluciones, en experimentos preeliminares. Finalmente, la que entrega mejores resultados es (12).

$$(p_i - c_i) \times \frac{\alpha_i}{\sum_{k \in I} \alpha_k} \tag{10}$$

$$\frac{p_i - c_i}{\mathrm{W}} \times \alpha_i \tag{11}$$

$$\frac{p_i - c_i}{\mathrm{W}} \times \frac{\alpha_i}{\sum_{k \in I} \alpha_k} \tag{12}$$

4.3. Configuración del experimento

MH es implementada utilizando el lenguaje de programación Python 3.11 y Gurobi 10 Todos los experimentos fueron ejecutados en un computador con procesador Intel® Core™ i7-8750H (CPU @ 2.20GHz × 12) con 12 unidades de procesamiento independiente (Core) y 24 GiB de RAM, bajo el sistema operativo Ubuntu (versión 22.04.3). El procesamiento de las 1600 instancias se realizó con un enfoque de procesamiento concurrente, asignado para cuatro hilos. Cada instancia se procesa en un hilo pero la metaheurística se procesa de manera secuencial en ese hilo. La condición de detención de MH es un tiempo límite de iteración por instancia, siendo este de 3000 segundos.

4.4. Medida de calidad de la solución

La medida de calidad de las soluciones se determina mediante la diferencia porcentual entre la solución generada por MH y la solución proporcionada por el solver exacto. Se define un criterio de evaluación denominado gap de optimalidad y se fija un límite del 5 % al igual que Baldo *et al.* (2023) para evaluar la solución. Este gap de optimalidad se utiliza

para evaluar la eficacia de la solución encontrada en comparación con la solución óptima esperada.

4.5. Resultados computacionales

En Tabla 3 se muestra un resumen de las instancias y de los resultados del algoritmo en las 1600 instancias. La primera columna muestra el tamaño de la instancia, la columna siguiente es el gap promedio de las 200 instancias correspondientes a cada tamaño, la tercera columna muestra el promedio general del valor de la función objetivo, la cuarta columna es el valor medio del error y la última columna es la cantidad de mejores resultados clasificados como resultados de buena calidad. De acuerdo con el criterio definido anteriormente (gap menor o igual a 5 %). Según el criterio del gap óptimo, se puede decir que MH tiene un buen rendimiento promedio en todos los tamaños de instancia, especialmente en las pequeñas y grandes. La heurística muestra el peor desempeño en las instancias medianas, se puede observar que el error medio en estas instancias es mayor y la cantidad de mejores soluciones encontradas son pocas.

Tabla 3: Resultados de MH.

Tamaño	Gap %	Media valor objetivo	Media error	N°mejores soluciones
100	5,00	3539,84	187,53	200
300	3,23	10468,13	314,28	46
500	2,68	15786,28	424,42	33
700	2,03	21744,88	442,38	14
900	1,81	27819,26	496,34	2
1100	0,84	32587,71	274,14	63
1300	0,84	38441,23	298,37	38
1500	0,58	45098,48	251,77	25

4.6. Comparación con el estado del arte

El desempeño de MH es comparado con los algoritmos de Baldo *et al.* (2023), HML y AG. Los valores utilizados para la comparación son de referencia, ya que los recursos computacionales usados son distintos a los utilizados para evaluar la metaheurística propuesta. Para realizar una mejor comparación, se deben evaluar bajo los mismos estándares computacionales.

La Tabla 4 muestra la comparación de la media del valor objetivo entre los distintos métodos utilizados: MH, AG y HML. La segunda columna muestra la media del valor

de la función objetivo para la metaheurística propuesta, la segunda columna es la media del valor de la función objetivo para HML, la tercera columna es la media del valor de la función objetivo para GA. Finalmente, las tres columnas siguientes representan el error medio porcentual para cada uno de los métodos (gap), que es calculado según la ecuación (10).

$$\text{gap} = \frac{\text{valor objetivo algoritmo} - \text{valor objetivo solver}}{\text{valor objetivo solver}} \times 100\,\% \tag{13}$$

Tabla 4: Resumen comparación de los resultados obtenidos por los métodos.

Tamaño	**Media FO**			**Gap %**		
	MH	**HML**	**GA**	**MH**	**HML**	**GA**
100	3539,78	3268,12	3295,20	5,00	2,53	1,71
300	10468,10	9901,19	9837,32	3,23	2,50	3,21
500	15786,17	15090,02	14879,81	2,68	1,84	3,19
700	21744,76	20906,23	20568,02	2,03	1,84	3,48
900	27819,20	26808,09	26269,35	1,81	1,92	4,00
1100	32587,71	31900,77	31175,11	0,84	1,20	3,61
1300	38441,22	37688,28	36718,47	0,84	1,19	3,79
1500	45098,43	44299,76	42795,21	0,58	1,18	4,67

Todos los métodos producen soluciones de buena calidad, manteniendo el gap promedio por debajo o igual al 5 %, como se aprecia en la Tabla 4. Es relevante destacar que, a pesar de las diferencias en las medias del valor de la función objetivo entre los métodos, la variación máxima es de 1000 unidades. Esta pequeña variación sugiere una similitud en los resultados obtenidos por los diferentes métodos. A pesar de las divergencias en las soluciones, la calidad general de los resultados es consistente entre los métodos evaluados.

En la Tabla 5, se profundiza en la comparación de la calidad de las soluciones. Con este fin, se introduce un criterio adicional: la diferencia entre el valor de la función objetivo obtenido por cada método y el valor de la función objetivo entregado por el solver exacto sea menor a 200 para considerarse una mejor solución. Las columnas de la tabla presentan la cantidad de instancias para las cuales se cumple este criterio para los métodos estudiados, o sea, en las que se obtiene una solución de calidad muy cercana a la solución del solver general. Este enfoque proporciona una visión detallada de la proximidad las soluciones de los diferentes métodos en relación con la referencia proporcionada por Gurobi.

El GA demuestra ser muy eficiente en la instancia más pequeña, pero su efectividad disminuye a medida que las instancias crecen. Por otro lado, la metaheurística propuesta destaca generando la mayor cantidad de soluciones que cumplen el criterio y mantiene un comportamiento estable a través de diferentes tamaños de instancias. Aunque HML supera a GA y muestra una mayor estabilidad, aun así, se considera que MH ofrece los mejores resultados. Es crucial considerar que todos los métodos enfrentan limitaciones en cuanto al tiempo usado. Por ende, si se prueban los métodos con un mayor tiempo disponible, los resultados podrían experimentar cambios significativos. Estas observaciones resaltan la importancia de considerar el factor tiempo al evaluar la eficacia de los algoritmos y sugieren posibles direcciones para investigaciones futuras.

Tabla 5: Cantidad de mejores soluciones obtenidas con cada métodos.

Tamaño	**Numero de mejores soluciones**		
	MH	**GA**	**HML**
100	129	200	188
300	109	25	94
500	87	0	81
700	89	0	34
900	87	0	17
1100	133	0	34
1300	125	0	22
1500	135	0	7

Capítulo 5

Conclusiones

Esta memoria de título propuso una metaheurística de búsqueda iterada para el PRKP. El algoritmo propuesto está basado en la estrategia de resolución de subproblemas, con el fin de encontrar los mejores candidatos a ser elegidos en la solución final, reduciendo así el espacio de búsqueda y mejorando las soluciones para el PRKP. La metaheurística propuesta es simple y requiere de escasos parámetros. Además, demuestra producir soluciones de buena calidad, dentro de los límites de tiempo asignados, y sus resultados son competitivos con las heurísticas de la literatura.

Los resultados obtenidos revelaron que en las instancias medianas se tiene peor rendimiento que en los otros métodos. Este hallazgo sugiere oportunidades de mejora, especialmente en la formación de subconjuntos, donde la refinación de procesos computacionales podría ofrecer mejores estadísticas sobre las variables, contribuyendo así a una solución más eficiente del problema.

Como trabajo futuro se propone la exploración de nuevas heurísticas o mejorar los enfoques existentes. Para el algoritmo propuesto existen diversas aristas que abordar. En primer lugar, se podría perfeccionar la lógica de la programación para ejecutar la metaheurística, optimizando los procesos de generación y evaluación de soluciones. Además, se podría aplicacar búsquedas locales después de la generación de soluciones factibles, proporcionando al solver de propósito general una solución inicial más refinada.

Por otro lado, podría ser interesante explorar a profundidad las técnicas de machine learning e incluso combinarlo con el algoritmo propuesto. Las características evaluadas en MH pueden ser utilizadas como variables del modelo de machine learning para estimar la factibilidad de una variable de ser seleccionada como parte de la solución final.

Bibliografía

Baldo, A., Boffa, M., Cascioli, L., Fadda, E., Lanza, C., y Ravera, A. (2023). The polynomial robust knapsack problem. *European Journal of Operational Research*, 305(3):1424–1434.

Ben-Tal, A. y Nemirovski, A. (2000). Robust solutions of linear programming problems contaminated with uncertain data. *Mathematical Programming*, 88(3):411–424.

Bertsimas, D. y Sim, M. (2003). Robust discrete optimization and network flows. *Mathematical Programming*, 98(1-3):49–71.

Billionnet, A., Faye, A., y Soutif, (1999). A new upper bound for the 0-1 quadratic knapsack problem. *European Journal of Operational Research*, 112(3):664–672.

Büsing, C., Goderbauer, S., Koster, A. M. C. A., y Kutschka, M. (2019). Formulations and algorithms for the recoverable -robust knapsack problem. *EURO Journal on Computational Optimization*, 7(1):15–45.

Eilon, S. (1987). Application of the knapsack model for budgeting. *Omega*, 15(6):489–494.

Gallo, G., Hammer, P. L., y Simeone, B. (1980). Quadratic knapsack problems.

Glover, F., Kochenberger, G. A., Alidaee, B., y Amini, M. (2002). Solving quadratic knapsack problems by reformulation and tabu search: Single constraint case. En *Series on Applied Mathematics*, pp. 111–121.

Hammaer, P. y Rader, D. J. (1997). Efficient methods for solving quadratic 0–1 knapsack problems. *INFOR: Information Systems and Operational Research*, 35(3):170–182.

Han, J., Lee, K., Lee, C., Choi, K., y Park, S. S. (2015). Robust optimization approach for a chance-constrained binary knapsack problem. *Mathematical Programming*, 157(1):277–296.

Kellerer, H. y Strusevich, V. A. (2008). Fully polynomial approximation schemes for a symmetric quadratic knapsack problem and its scheduling applications. *Algorithmica*, 57(4):769–795.

Létocart, L., Nagih, A., y Plateau, G. (2012). Reoptimization in lagrangian methods for the 0–1 quadratic knapsack problem. *Computers and Operations Research*, 39(1):12–18.

Mirshekarian, S. y Šormaz, D. (2018). Machine learning approaches to learning heuristics for combinatorial optimization problems. *Procedia Manufacturing*, 17:102–109.

Monaci, M., Pferschy, U., y Serafini, P. (2013). Exact solution of the robust knapsack problem. *Computers and Operations Research*, 40(11):2625–2631.

Nobibon, F. T. y Leus, R. (2013). Complexity results and exact algorithms for robust knapsack problems. *Journal of Optimization Theory and Applications*, 161(2):533–552.

Poss, M. (2018). Robust combinatorial optimization with knapsack uncertainty. *Discrete Optimization*, 27:88–102.

Rezoug, A., Bader-El-Den, M., y Boughaci, D. (2021). Application of supervised machine learning methods on the multidimensional knapsack problem. *Neural Processing Letters*, 54.

UNIVERSIDAD DE CONCEPCIÓN – FACULTAD DE INGENIERÍA
RESUMEN DE MEMORIA DE TÍTULO

Departamento: Departamento de Ingeniería Civil Industrial

Carrera: Ingeniería Civil Industrial

Nombre del memorista: Daniel Antonio Castillo Molina

Título de la memoria: Una metaheurística para resolver el polynomial robust knapsack problem

Fecha de presentación oral: 20 de Marzo del 2024

Profesor(es) Guía: Carlos Contreras Bolton

Profesor(es) Revisor(es): Lorena Pradenas

Concepto:

Calificación:

Resumen
El polinomial robust knapsack problem (PRKP) es una variante del clásico knapsack problem (KP), que implica seleccionar un conjunto de elementos para maximizar la utilidad, cumpliendo con restricciones de capacidad. En el PRKP, los elementos tienen un beneficio independiente y uno dependiente de otros elementos seleccionados, llamado sinergia, reflejando situaciones del mundo real como proyectos que se benefician mutuamente. Además, los costos asociados a cada elemento son inciertos y varían en un rango determinado. Dada su relevancia en diversos campos, especialmente en inversiones, la investigación del PRKP es significativa. Esta memoria de título presenta una heurística para resolver el PRKP, utilizando una metodología que genera soluciones parciales mediante muestreo aleatorio y las resuelve con un solver de propósito general. El algoritmo propuesto se validó en 1600 instancias y se comparó con otros métodos del estado del arte, mostrando un rendimiento competitivo, con diferencias de valor en la función objetivo igual o inferiores al 5% en comparación con el solver de propósito general.