EDUCACIÓN
SECRETARÍA DE EDUCACIÓN PÚBLICA

TECNOLÓGICO NACIONAL DE MÉXICO

INSTITUTO TECNOLÓGICO DE CIUDAD MADERO

DIVISIÓN DE ESTUDIOS DE POSGRADO E INVESTIGACIÓN

MAESTRÍA EN CIENCIAS DE LA COMPUTACIÓN

"POR MI PATRIA Y POR MI BIEN"

TESIS

ALGORITMO EVOLUTIVO PARA LA SOLUCIÓN DEL PROBLEMA DE CARTERA DE PROYECTOS CON INTERVALOS Y MUCHOS OBJETIVOS

Que para obtener el Grado de
Maestro en Ciencias de la Computación

Presenta
Ing. Lorena Rocío Rosas Solórzano
G13071208

Director de Tesis
Dra. Claudia Guadalupe Gómez Santillán

Co- director de Tesis
Dr. Nelson Rangel Valdez

Cd. Madero, Tamaulipas Junio 2021

Cd. Madero, Tam. 18 de mayo de 2021

OFICIO No. : U.028/21
ASUNTO: AUTORIZACIÓN DE IMPRESIÓN DE TESIS

C. LORENA ROCIO ROSAS SOLORZANO
No. DE CONTROL G13071208
P R E S E N T E

Me es grato comunicarle que después de la revisión realizada por el Jurado designado para su Examen de Grado de Maestría en Ciencias de la Computación, se acordó autorizar la impresión de su tesis titulada:

"ALGORITMO EVOLUTIVO PARA LA SOLUCIÓN DEL PROBLEMA DE CARTERA DE PROYECTOS CON INTERVALOS Y MUCHOS OBJETIVOS"

El Jurado está integrado por los siguientes catedráticos:

PRESIDENTE:	DR. HECTOR JOAQUÍN FRAIRE HUACUJA
SECRETARIO:	DRA. MARÍA LUCILA MORALES RODRÍGUEZ
VOCAL:	DRA. CLAUDIA GUADALUPE GÓMEZ SANTILLÁN
SUPLENTE:	DR. NELSON RANGEL VALDEZ
DIRECTOR DE TESIS:	DRA. CLAUDIA GUADALUPE GÓMEZ SANTILLÁN
CO-DIRECTOR DE TESIS:	DR. NELSON RANGEL VALDEZ

Es muy satisfactorio para la División de Estudios de Posgrado e Investigación compartir con usted el logro de esta meta. Espero que continúe con éxito su desarrollo profesional y dedique su experiencia e inteligencia en beneficio de México.

A T E N T A M E N T E
Excelencia en Educación Tecnológica®
"Por mi patria y por mi bien"®

MARCO ANTONIO CORONEL GARCÍA
JEFE DE LA DIVISIÓN DE ESTUDIOS DE POSGRADO E INVESTIGACIÓN

EDUCACIÓN | TECNOLÓGICO NACIONAL DE MÉXICO
INSTITUTO TECNOLÓGICO DE CIUDAD MADERO
DIVISIÓN DE ESTUDIOS DE POSGRADO E INVESTIGACIÓN

c.c.p.- Archivo
MACG 'mdcoa'

Declaración de Originalidad

Declaro y prometo que este documento de tesis es producto de mi trabajo original y que no infringe los derechos de terceros, tales como derechos de publicación, derechos de autor, patentes y similares.

Además, declaro que en las citas textuales que he incluido (las cuales aparecen entre comillas) y en los resúmenes que he realizado de publicaciones ajenas, indico explícitamente los datos de los autores y las publicaciones.

Además, en caso de infracción de los derechos de terceros derivados de este documento de tesis, acepto la responsabilidad de la infracción y relevo de ésta a mi director y codirectores de tesis, así como al Tecnológico Nacional de México campus Ciudad Madero y sus autoridades.

09 de Junio de _2021_, Cd. Madero, Tamps.

_____

Ing. Lorena Rocío Rosas Solórzano

Resumen

La selección de la cartera de proyectos (PPS) es un importante problema de decisión que enfrenta cualquier organización. PPS decide cómo invertir recursos en proyectos sujetos a un proceso de decisión, influenciado por múltiples criterios en conflicto. El compromiso de la cartera con el bienestar de la organización, tiene una incertidumbre que afecta directamente las preferencias del tomador de decisiones (DM).

MOEA/D es un enfoque bien conocido para abordar problemas de optimización multicriterio, y aún está abierto al desarrollo de estrategias para manejar la incertidumbre en su proceso de búsqueda. Este trabajo se propone a I-MOEA/D, un nuevo método basado en un enfoque MOEA/D, para lidiar con la incertidumbre del DM en los costos y beneficios de los proyectos de las carteras.

Las características novedosas propuestas incluyen:

a) Manejar un gran número de objetivos, realizando experimentaciones con 2,3,4,8,9,13 y 15 objetivos, haciendo uso de instancias artificiales, creadas mediante un generador de pesos, con uso de funciones de logaritmo natural y un generador de instancias de cartera de proyectos multiobjetivo con intervalos.

b) Un método para generar la población inicial, el cual es una estrategia de inicialización para el algoritmo I-MOEA/D, de tal modo que hace uso de intervalos presentes en la información de las carteras, siendo un proceso, que elige un proyecto, siempre y cuando el valor aleatorio uniforme se encuentre por debajo de una selección de umbral predefinida (establecido en 0.5 para este trabajo de investigación)

c) Manejar la incertidumbre de recursos, costos y beneficios a través de intervalos.

Un experimento comparó I-MOEA / D con el algoritmo I-NSGA-II del estado del arte, en instancias con dos a quince objetivos. Los resultados demuestran la competitividad de I-MOEA/D al mejorar la calidad de la solución de I-NSGA-II en la mayoría de los casos.

Agradecimientos

Al concluir esta etapa maravillosa de mi vida, quiero extender un profundo agradecimiento a quienes hicieron posible este sueño, aquellos que junto a mí brindaron su apoyo, siendo inspiración y fortaleza a mi vida. Esta mención es en especial para Dios, mi familia, profesores y amigos.

Agradezco especialmente a la Dra. Claudia Guadalupe Gómez Santillán, directora de mi proyecto de investigación y al Dr. Nelson Rangel Valdez por sus valiosas recomendaciones y apoyo incondicional, además de ser mis guías durante esta etapa profesional. De igual manera agradezco a mi tutora, la Dra. María Lucila Morales Rodríguez, al Dr. Héctor Joaquín Fraire Huacuja y a la Dra. Laura Cruz Reyes por sus opiniones y correcciones en este proyecto.

Agradezco el apoyo de las instituciones públicas que hicieron posible este trabajo: el Consejo Nacional de Ciencia y Tecnología (CONACYT), el Tecnológico Nacional de México y el Instituto Tecnológico de Ciudad Madero.

Tabla de contenido

LISTA DE TABLAS

Lista de figuras

CAPITULO 1

Introducción

En la actualidad, la cantidad de información que se genera y administra en las organizaciones crece exponencialmente, lo cual hace que las empresas e instituciones manipulen grandes cantidades de información en los diversos departamentos, de tal manera que cuando se requiere tomar una decisión, se tiene que hacer un análisis de grandes cantidades de información haciendo difícil la toma de decisiones.

Olson en (Olson, 2010) menciona que son muchos los investigadores que han definido la toma de decisiones como una actividad esencial y central en las organizaciones. Para llevar a cabo la toma de decisiones en las organizaciones tradicionalmente se forman carteras de proyectos las cuales contienen un subconjunto de los proyectos.

Carazo en (Carazo, y otros, 2010) define un proyecto como un proceso temporal, único e irrepetible que persigue un conjunto específico de objetivos, los cuales al combinarse impactaran en la misión y visión de las organizaciones, entonces una cartera es definida por Fox & Baker en (Fox & Baker, 1985) como: “conjunto de proyectos que, llevados a cabo en un determinado periodo de tiempo, comparten una serie de recursos y entre los que pueden existir relaciones de complementariedad, incompatibilidad y sinergias producidas por compartir costos y beneficios derivados de la realización de más de un proyecto a la vez”.

Entonces al formar carteras algunas de ellas aportaran mayor beneficio a las organizaciones es por eso que tomar una “buena” o “mala” decisión compromete en gran medida el bienestar de la organización y es de suma importancia realizar un análisis para toma de decisiones de cartera de proyectos.

El análisis de la decisión de carteras (Portfolio Decision Analysis, PDA) puede ser definido como el conjunto de teorías, métodos y prácticas que ayudan al tomador de decisiones a seleccionar un subconjunto de un conjunto muy grande de proyectos a través de modelación matemática tomando en cuenta restricciones relevantes, preferencias y la incertidumbre (Salo, Keisler, & Morton, 2011). La incertidumbre es un rango o inexactitud presente en la información (Larousse, 2016).

Esta tesis es una aportación en la solución del problema de cartera de proyectos aplicando un algoritmo evolutivo multiobjetivo basado en descomposición conocido como MOEA-D (por sus siglas en inglés) introduciendo matemáticas de intervalos y así mejorar la calidad de las soluciones reportadas en la literatura. Un intervalo es sinónimo de incertidumbre y se han definido para los recursos de la organización, por ejemplo, beneficios, costos, tiempos, sinergias, entre otros. La metodología, incluye las preferencias del tomador de decisiones (DM); el DM es el responsable de seleccionar la cartera que mejor satisface los objetivos de la organización.

En las secciones de este capítulo, se presenta un panorama general de la tesis, presentando los antecedentes del proyecto, la justificación, los objetivos generales y específicos del trabajo, alcances y limitaciones, finalizando con la organización del documento.

1.1 Antecedentes

Esta investigación forma parte de un proyecto de red OPTISAD que está constituida por investigadores de la Universidad Autónoma de Sinaloa (UAS), la Universidad Autónoma de Nuevo León (UANL), la Universidad de Occidente (UdeO), Universidad Autónoma de Coahuila (UAC) y el Instituto Tecnológico de Ciudad Madero (ITCM). Los cuerpos académicos de estas instituciones han colaborado en el desarrollo de métodos e instrumentos enfocados a la optimización de cartera de proyectos con el objetivo de propiciar su integración en una metodología general.

Resolver el problema de cartera de proyectos requiere de una convergencia multidisciplinar en disciplinas matemáticas (estadística, programación matemática, lógica borrosa), ingenieriles (sistemas, computación), híbridas (análisis de la decisión, inteligencia artificial, análisis multicriterio) y administrativas. Debido a esto, surge el acercamiento entre los investigadores de las instituciones educativas participantes, bajo la creación de una red, denominada "Red temática para apoyo a la decisión y optimización inteligente de sistemas complejos y de gran escala", que está compuesta por los siguientes cuerpos académicos: "Sistemas de Apoyo a la Toma de Decisiones" de la UAS, "Ingeniería en Sistemas" de la UANL, "Administrativos" de la UdeO y "Optimización Inteligente" del ITCM.

Este proyecto de red pertenece a una de las líneas de investigación qué está enfocada en la optimización de cartera con muchas funciones objetivo utilizando procedimientos metaheurísticos, análisis y decisión multicriterio. El presente trabajo de investigación, propone el tratamiento de incertidumbre inmersa en el problema de cartera de proyectos utilizando matemática de intervalos empleando un algoritmo evolutivo multiobjetivo. El objetivo de la investigación es continuar con el desarrollo de instrumentos que apoyen a conseguir las metas planteadas por el grupo de investigadores complementando y aportando los trabajos desarrollados por investigadores de la red.

1.2 Justificación

Debido a que la información en las organizaciones se genera en grandes cantidades, en estos tiempos cada vez es más común tener problemas con el análisis de la información para la toma de decisiones. Por tal motivo es necesario la ayuda de herramientas capaces de brindar apoyo en la toma de decisiones y así cumplir con los objetivos de las organizaciones.

La mayoría de los trabajos que abordan el problema de selección de cartera de proyectos no toman en cuenta la imprecisión inmersa en los procesos administrativos de las organizaciones, es decir establecen un costo fijo para los proyectos y en el momento de

poner en marcha el proyecto el costo estimado con anterioridad cambio. Esto implica que las organizaciones necesiten hacer ajustes en sus presupuestos y en algunos casos no cumplir con los objetivos anuales establecidos.

Actualmente existe un gran número de enfoques en el estado del arte que han sido propuestos para resolver el problema de estudio, por ejemplo, en (Rivera,2015) (Salas, 2019) (Balderas 2018), algunos enfoques abordan: el estudio del problema con sinergia y apoyo parcial (Rivera, 2015) y aplican estrategias hibridas de técnicas de algoritmos exactos y metaheurísticos; otros enfoques abordan el problema de estudio con pocos objetivos (dos objetivos) e incluyen incertidumbre en algunos factores del problema, un ejemplo de este enfoque es el trabajo de Salas (Salas,2019). Por otro lado, en Balderas (2018) toma en consideración la incertidumbre e incorpora las preferencias del tomador de decisiones usando un enfoque de sobre clasificación basado en intervalos con la finalidad de dirigir la búsqueda hacia la región de interés de la frontera de Pareto, cabe señalar que también trabaja con pocos objetivos.

Dentro del grupo de trabajo se han ocupado otras metaheurística, por tal motivo se propuso trabajar con MOEA/D aplicando incertidumbre, siendo que los trabajos que han abordado el problema con incertidumbre tienen pocos objetivos (I-NSGAII).

1.3 Objetivos

A continuación, se presentan el objetivo general y los objetivos específicos planteados para este trabajo de investigación.

1.3.1 Objetivo general

Resolver el problema de cartera de proyectos con intervalos y muchos objetivos, mediante un algoritmo evolutivo multiobjetivo que logre una calidad en las soluciones competitivas con las del estado del arte.

1.3.2 Objetivos específicos

Los objetivos específicos que favorecen el cumplimiento del objetivo general son:

1. Analizar el desempeño de metaheurísticas del estado del arte que resuelven el problema de cartera de proyectos con intervalos y muchos objetivos.
2. Desarrollar una solución para el problema de cartera de proyectos con intervalos y muchos objetivos a través del algoritmo MOEA/D.
3. Desarrollar indicadores de calidad de desempeño para el algoritmo evolutivo MOEA/D.

1.4 Alcances y limitaciones

A continuación, se enlistan los alcances y limitaciones de este trabajo de investigación:

a. El problema resuelto es cartera de proyectos multiobjetivo.
b. Los valores que reflejen incertidumbre, es modelado utilizando intervalos.
c. La cantidad de objetivos es hasta 15 objetivos como máximo.
d. El alcance de la investigación está reducido al tratamiento de incertidumbre en el presupuesto, costo, objetivos y beneficio del problema de cartera de proyectos.

1.5 Organización del documento

Este documento está estructurado como sigue: El capítulo 2 proporciona el marco teórico, el cual conlleva los conceptos, que son el soporte teórico de la investigación desarrollada en este trabajo de investigación. El capítulo 3 describe el estado del arte del problema de investigación, considerando el problema de investigación, la complejidad y el estado del arte. En el capítulo 4 muestra la metodología de solución, dónde se describe la adaptación de intervalos en el problema de selección de cartera de proyectos multiobjetivo y la descripción de la propuesta I-MOEA/D. El capítulo 5 describe el experimento realizado y los resultados para validar I-MOEA/D; además proporciona el análisis que demuestra las ventajas de la estrategia propuesta. El capítulo 6 resume las principales conclusiones del trabajo de investigación.

CAPITULO 2

Marco Teórico

En esta sección se introducen conceptos relacionados con la teoría que sustenta este trabajo de investigación. Principalmente, se describe sobre el concepto fundamental de optimización y sus variantes, además de las estrategias de solución para problemas de optimización. Por consiguiente, se verá cómo funcionan los algoritmos evolutivos adentrando en los algoritmos genéticos. Después se describen los operadores genéticos de selección, cruza y mutación, por consiguiente, se definen los algoritmos evolutivos multiobjetivo conocido como MOEA/D (Multiobjetive Evolutionary Alghorithm based on Descomposition) y NSGA-II (*Non-dominated Sorting Genetic Algorithm II*), continuando con las pruebas estadísticas detallando la prueba de signos de Wilcoxon. Concluyendo con conceptos fundamentales de la matemática de intervalos.

2.1 Problemas de optimización

Un problema de optimización consiste en minimizar o maximizar el valor de una variable, en el lenguaje cotidiano "optimizar" significa poco más que mejorar algo, sin embargo, (Duarte, 2007) explica la optimización como "el proceso de intentar encontrar la mejor solución posible a un problema de optimización, generalmente en un tiemplo limitado".

Un problema de optimización se conforma de tres elementos importantes:

- **Variables de decisión:** contiene los valores que se modifican para resolver el problema.
- **Función objetivo:** Se expresa en términos de las variables de decisión, y el resultado de su evaluación es el que se desea optimizar (maximizar o minimizar).

- **Restricciones:** expresada en forma de ecuaciones de igualdad (ecuación 1.3) o desigualdad (ecuación 1.2), se deben cumplir o satisfacer para que la solución sea considerada factible, es decir, sea válida.

En un problema de minimización, dada una función $f: \mathbb{X} \in \mathbb{R}^n \rightarrow \mathbb{R}$ se trata de hallar el vector $x \in \mathbb{R}$ que minimice dicha función como se indica en la ecuación 1.1.

$$\min_x f(x) \tag{1.1}$$

Sujeto a:

$$g_i(x) \leq 0 \qquad i = 1,2,3,\dots,m \tag{1.2}$$

$$h_j(x) = 0 \qquad j = 1,2,3,\dots,p \tag{1.3}$$

Donde f es un vector de k funciones objetivo $f = (f_1, \dots, f_n)$ para la cual se desea obtener el valor mínimo o máximo según sea la función objetivo (López, 2014), x es el vector de solución $x = (x_1, \dots, x_r)^t$ de *r* variables, *n* es el número de restricciones de desigualdad.

Las restricciones definidas acotan el espacio de soluciones, dividiendo el espacio de búsqueda en dos regiones:

- **Soluciones Factibles:** Aquellos elementos del espacio de búsqueda que cumplen con todas las ecuaciones de restricción.
- **Soluciones No Factibles:** Aquellos elementos del dominio de la función que no cumplen por lo menos una de las restricciones del problema.

Cabe mencionar, que si solo existe una función objetivo se considera que es un problema de optimización mono-objetivo, en caso de que sean dos o más objetivos, se le denomina problema de optimización multiobjetivo (Cagnina, 2010).

2.2 Problemas de optimización multiobjetivo

Un problema de optimización multiobjetivo (MOP) difiere de un problema mono-objetivo, dado que los primeros requieren la optimización simultanea de más de una función objetivo en paralelo (López, 2014). Un MOP es definido por Osyczka (1985) como: "Encontrar un vector de variables de decisión que satisfaga las restricciones dadas y optimice un vector de funciones cuyos elementos representan las funciones objetivo" y se denota de manera general por medio de la ecuación 1.4, donde *x* son las variables de decisión, pertenecientes a un vector $\vec{x} = \langle x_1, x_2, \dots, x_n \rangle$ dentro de un espacio de soluciones factibles R_F, $f(x)$, donde cada f_i representa una de las *n* funciones objetivos a minimizar o maximizar, encontrando el vector $\vec{x}$ que satisfaga las *m* restricciones de desigualdad (ecuación 1.5) y las *p* restricciones de igualdad (ecuación 1.6):

$$\underset{x \in R_F}{f(x)} = \{f_1(\vec{x}), f_2(\vec{x}), \dots, f_n(\vec{x})\} \quad (1.4)$$

$$g_i(\vec{x}) = 0, i = 1, \dots, m \quad (1.5)$$

$$h_i(\vec{x}) \leq 0, i = 1, \dots, p \quad (1.6)$$

Cuando los objetivos se encuentran en conflicto en la optimización multiobjetivo, se busca un compromiso entre las soluciones en lugar de única solución, recurriendo frecuentemente al uso de estrategias de optimización.

2.3 Estrategias de solución para problemas de optimización

En la actualidad existen diversas propuestas algorítmicas para la solución de problemas, un algoritmo puede ser de solución exacta o aproximada. Un algoritmo exacto garantiza la obtención de la solución óptima, por el contrario, un algoritmo aproximado no puede garantizar la obtención del óptimo (Woeginger, 2003). Los algoritmos se clasifican en exactos y de aproximación.

Métodos exactos: Los métodos exactos garantizan una solución óptima, siempre que ésta exista; sin embargo, en problemas grandes o muy complejos podría hacerse inviable ya que

el tiempo de cómputo crece en forma exponencial con el tamaño del problema (Morillo, Moreno & Díaz, 2014).

Métodos aproximados: Los métodos aproximados construyen una solución que no puede garantizar que sea optima, y están diseñados para resolver problemas difíciles de optimización combinatoria (J.P. Kelly, 1996). Un método metaheurístico es un algoritmo aproximado y este término fue introducido por Glover (1986), dentro de esta clasificación se destacan los algoritmos evolutivos.

2.4 Algoritmos evolutivos

Los algoritmos evolutivos (EA) son métodos robustos de búsqueda, que permiten tratar problemas de búsqueda, que permiten tratar problemas de optimización, donde el objetivo es encontrar un conjunto de parámetros de minimizan o maximizan una función de adaptación. Estos algoritmos operan con una población de individuos $P(t) = \{x_1^t, \dots, x_n^t\}$ para la iteración *t*, donde cada individuo x_i representa un punto de búsqueda en el espacio de soluciones (Estévez, 1997).

La estructura del algoritmo evolutivo básico tiene la siguiente forma (Algoritmo 1):

Algoritmo 1. Algoritmos evolutivos

```
0.  t=0
1.  Inicializar P(t)
2.  Evaluar P(t)
3.  while (no condición de paro) do
4.      t=t+1
5.      Padres= Selección P(t)
6.      HijoC= Cruza (Padres)
7.      HijoM= Mutación (HijoC)
8.  end
```

Primeramente se genera una población aleatoria (línea 1) y es evaluada (línea 2) la cual debe cumplir ciertas restricciones planteadas por el problema y en cada iteración *t* se selecciona un subconjunto de la población actual (línea 5) para generar nuevos hijos

mediante recombinación aplicando una alteración de cruza (línea 6) y mutación (línea 7) a los individuos que fueron seleccionados y si cumplen con el criterio de optimización devuelve la mejor solución encontrada, en caso contrario vuelve a realizar la selección y alteración de los individuos (Algoritmo 1).

Los algoritmos evolutivos presentan ciertas ventajas en la resolución de problemas de optimización (Goldberg, 1989):

- Operan sobre una población (o conjunto de soluciones) lo que evita que la búsqueda se quede atascada en óptimos locales.
- No requieren conocimiento previo sobre el problema a resolver.
- Pueden combinarse con otras técnicas de búsqueda para mejorar su desempeño.
- Permiten su paralelización de forma sencilla. Son conceptualmente fáciles de implementar y usarse.
- Utilizan operadores probabilísticos, en comparación con las técnicas tradicionales que utilizan operadores determinísticos.
- Generalmente pueden auto adaptar sus parámetros.

2.5 Algoritmo genético

Dentro de los evolutivos, se encuentra el algoritmo genético (AGs), el cual busca el espacio de soluciones de una función a través de la estrategia de la supervivencia del más apto.

En los seres humanos, el ácido desoxirribonucleico (ADN) se almacena de una manera estructurada. A esta estructura se le denomina cromosoma. Si se compara lo biológico con la vida artificial, es decir los AGs, un cromosoma indica una posible solución del problema, que por lo general se presenta como una cadena de bits (Fig. 1), que puede ser real o binaria. Un alelo en una cadena de bits será un 0 o 1, para un dominio discreto el número de alelos aumenta en dependencia del problema (Moré, 2017).

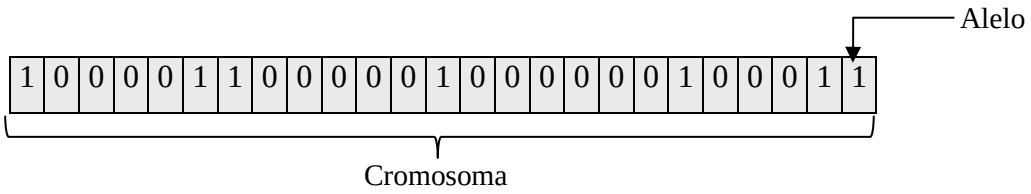

Figura 1. Ejemplo de solución binaria

Estos algoritmos comúnmente se utilizan para solucionar problemas lineales y no-lineales, explorando todas las áreas posibles, además de crear nuevas soluciones aplicando operadores genéticos a los elementos individuales de la población.

2.6 Operadores genéticos

Los principales operadores de los algoritmos genéticos son la selección, la cruza y la mutación. Los operadores genéticos a menudo dependen del problema y son de importancia crítica para un uso exitoso en problemas prácticos (Wilfried, Xing & Dezhhen, 2020).

La selección de padres para producir generaciones sucesivas, su objetivo es seleccionar con más frecuencia a los individuos más aptos para reproducirse; mientras que la cruza, es responsable de la recombinación de la estructura, intercambiando información entre los cromosomas de apareamiento, por otro lado, la mutación es necesaria porque, aunque la selección y la cruza buscan y recombinan eficazmente las nociones de extensión, ocasionalmente pueden perder algún material genético potencialmente útil (Debasish, Abhijit & Aparajita, 2020).

2.6.1 Selección

En términos biológicos los individuos más adaptados en un ecosistema o medio ambiente tienen más probabilidad de sobrevivir y de procrear en mayor medida. Sin embargo, en las técnicas metaheurísticas, seleccionar siempre aquellos individuos cuya función de aptitud fuera mejor, correríamos el riesgo de converger a un óptimo local de forma prematura. Por esta razón se hace necesario que la población del AG se someta a un proceso de selección

aleatorio que debe tender a favorecer en cierta medida a los individuos más adaptados. Este proceso se puede realizar de diferentes formas (Moré, 2017):

- Selección por ruleta (Jong, 1975).
- Selección de estado uniforme (Jong, 1975)
- Selección sobrante estocástico (Brindle, 1981)
- Selección universal estocástica (Baker, 1987)
- Selección muestreo determinístico (Golberg, 1989)
- Selección escalamiento sigma (Mitchell, 1989)
- Selección de Bolzmann (Golbert, 1990)
- Selección por jerarquías lineales (Baker, 1985)
- Selección por torneo (Whitley, 1989)

2.6.2 Cruza

La cruza es un operador que forma un nuevo cromosoma combinando partes de cada uno de sus cromosomas padres (Coello, 2011). Los cromosomas se alinean, para después fraccionarse con la finalidad de intercambiar fragmentos entre sí. En computación evolutiva este proceso se lleva a cabo generalmente intercambiando segmentos de cadenas entre dos individuos de la población. Dentro de las técnicas de cruza más usadas están (Deb, 1991):

- Cruza de un punto (Holland,1975)
- Cruza de dos puntos (De Jong, 1975)
- Cruza uniforme (Syswerda, 1989)
- Cruza acentuada (Schaffer, 1987)

2.6.3 Mutación

La mutación se basa en un operador básico, que brinda aleatoriedad a los individuos de una población. Si bien el operador de cruce se encarga de hacer una búsqueda en el espacio de posibles soluciones, es el operador de mutación el encargado de aumentar o reducir el

espacio de búsqueda en un algoritmo genético y de proporcionar cierta variabilidad genética de los individuos (Sosa, 2013). Existen diversos operadores de mutación y ocurre con muy baja probabilidad, generalmente menor al 1% del total de genes de los individuos de la población (Moré, 2017), algunas de las técnicas de mutación son (Coello, 2012):

- Mutación por inserción
- Mutación por desplazamiento
- Mutación por intercambio reciproco
- Mutación heurística

2.7 Algoritmos evolutivos para optimización multiobjetivo

Los EA se han consagrado como los métodos predilectos para solucionar problemas de optimización multiobjetivo. Las principales razones para utilizar algoritmos evolutivos para optimización multiobjetivo son (Deb, 2000):

1. Capaces de generar como salida un conjunto de soluciones compromiso en una sola ejecución del algoritmo.
2. Son insensibles a la forma de las funciones objetivo

En este trabajo de investigación nos enfocamos en dos algoritmos genéticos para encontrar la población externa, que son las carteras no dominadas. Estos algoritmos son el NSGA-II y MOEA/D.

2.7.1 Non- dominated Sorting Genetic Algorithm II

El NSGA-II (Algoritmo 2) es una propuesta mejorada del NSGA (Deb, 1994). El NSGA-II resuelve problemas preferentemente de dos o tres objetivos (Rivera, 2014), debido a su simplicidad y efectividad en la solución con pocos objetivos. El algoritmo NSGA-II, crea una población de padres P_0 de tamaño N (línea 0), la población se ordena de acuerdo con los niveles de no-dominancia (línea 3). Cada individuo se le asigna un valor de la aptitud en

base a la dominancia (línea 4). La población de hijos (Q_{t+1}) se genera mediante operadores genéticos de selección, cruza y mutación.

El procedimiento elitista para una generación $t \geq 1$ se realiza de la siguiente manera: se combina la población de padres e hijos, obteniendo una población de tamaño *2N*. La nueva población se ordena de acuerdo con la dominancia en diversos frentes (Deb, 2002). La nueva población se forma con los individuos que forman parte del frente 0, frente 1 y así sucesivamente, hasta que el número de frentes sea igual o mayor a N (Balderas, J., 2018).

Las soluciones que cuentan con igual valor de dominancia formaran parte de un mismo frente (línea 3). La nueva población es formada seleccionando los individuos que pertenezcan a *F1*, y así sucesivamente, hasta que P_{T+1} sea igual o mayor a *N* (línea 6); los individuos restantes son rechazados.

Algoritmo 2. NSGA-II (Balderas, 2018)

Entrada: Iter*MAX*, *N*
Salida: *F0* de la última iteración del algoritmo

0. **Inicializar**: $P \leftarrow$ **PoblaciónAleatoria**(*N*), $Q \leftarrow \emptyset$
1. F**or** *T*=1 **to** Iter*MAX* **do**
2. RT $\leftarrow$ PT $\cup$ QT
3. *F* $\leftarrow$ **fast-non-dominated-sort**(*RT*)
4. i $\leftarrow$ 0, P*T+1* $\leftarrow \emptyset$
5. **Mientras** $|PT+1|+|Fi| \leq N$
6. **crowding-distance-assignment**(*Fi*)
7. $PT+1 \leftarrow PT+1 \cup Fi$
8. i $\leftarrow$ i+1
9. F**in mientras**
10. **crowding-distance-sort**(*Fi*, $\prec n$) //ordenamiento ascendente
11. $PT+1 \leftarrow PT+1 \cup Fi$ [1: N – $|PT+1|$]
12. $QT+1 \leftarrow$ **generar-nueva-población** (*PT+1*)
13. **fin for**
14. **regresa** *F0*

2.7.2 Multiobjective Evolutionary Algorithm based on decomposition

MOEA/D es una técnica propuesta por Zhang y Li en (Zhang y Li, 2007). Este algoritmo consiste en descomponer un problema de optimización multiobjetivo en varios

subproblemas de optimización escalar y los optimiza simultáneamente. Cada subproblema se optimiza utilizando información de sus varios subproblemas vecinos, lo que hace que MOEA/D tenga una menor complejidad computacional en cada generación (Algoritmo 3).

Algoritmo 3. MOEA/D

Input:

- *MOP*= Problema de optimización Multiobjectivo
- *N*= Tamaño de población
- *p*=Numero de proyectos
- *m*= Numero de objetivos
- *T= Tamaño del vecindario de los vectores de peso*
- *EvaluaciónMaxima*= Numero de generaciones

Output:

EP= Población externa

0. *W* = LeerVectorDePesos()
1. *EP*= ϕ
2. *Calcular_Distancia_Euclidiana* (W)
3. *OrdenarVector* ()
4. Población=*GenerarPoblaciónInicial*()
5. *Inicializar_Z (Población)*
6. Generaciones=0;
7. **While**(Generaciones < *EvaluaciónMaxima*)
8. *i*=1
9. **For each** $i \in N$ **do**
10. [p_1, p_2] =Selección (Population, *B(i)*, *T*)
11. *descendencia*=Cruza(*p*, [p_1,p_2])
12. *descendencia* =Mutación(*p*, *descendencia*)
13. *descendencia*=Improvement (*descendencia*)
14. *ActualizarZ* ($\vec{z}$, *f*(*descendencia*))
15. *ActualizarEP* (EP, *f*(*descendencia*))
16. **end**
17. Generations++
18. **end**

MOEA/D establece que *EP* debe iniciar vacía (línea 1). Después se calcula la distancia euclidiana entre los vectores de peso para conformar *T* (línea 2 y 3). Por consiguiente, se genera una población inicial aleatoria (línea 4) y se inicializa el vector *z*, tomando el valor máximo de cada objetivo de entre todas las carteras de la población inicial (línea 5).

Para realizar la actualización de la población, se realiza un ciclo desde 1 hasta N (línea 9 a 16), seleccionando dos carteras de la población inicial y aplicando métodos de mutación y cruza. Para validar estos procesos, es necesario hacer comparaciones entre los resultados para demostrar con significancia estadística que el proceso es reproducible, por tal motivo se recurre a las pruebas estadísticas.

2.8 Pruebas estadísticas

El hecho de querer comparar las medias de estas muestras hace que tengamos que lidiar con pruebas estadísticas. Para poder aplicar cada prueba, existen diversas hipótesis y condiciones que deben cumplir los datos para que los resultados de la prueba sean fiables. Pero en muchas ocasiones esta hipótesis no resulta válida, y en otras no es adecuada y no resulta fácil de comprobar, por tratarse de muestras pequeñas.

Las pruebas estadísticas, se dividen en dos conjuntos: las paramétricas y las no paramétricas. Las pruebas paramétricas solamente se pueden utilizar si los datos muestran una distribución normal. (Flores, Miranda, & Villasís, 2017) y las no paramétricas son las que, a pesar de basarse en determinadas suposiciones, no parten de la base de que los datos analizados adoptan una distribución normal (Scientific European Federation of Osteopaths, 2014).

Cuando la distribución de datos cuantitativos no sigue una distribución normal existen diferentes pruebas estadísticas con las que se comparan las medianas. Para la comparación de grupos independientes se debe emplear U-de Mann-Withney, por otro lado, la prueba de Wilcoxon se utiliza para comparar un grupo antes y después, es decir, para muestras relacionadas (Flores, Miranda, & Villasís, 2017).

2.8.1 Prueba de rangos con signo de Wilcoxon

La prueba de Wilcoxon es una prueba no paramétrica para comparar el rango medio de dos muestras relacionadas y determinar si existen diferencias entre ellas. Se utiliza como alternativa a la prueba t de Student cuando no se puede suponer la normalidad de las

muestras. Esta prueba de comparación de dos muestras relacionadas no necesita una distribución especifica (Scientific European Federation of Osteopaths, 2019). El procedimiento para aplicar la prueba de Wilcoxon con muestras correlacionadas es:

1. Calcular diferencia entre dos muestras D_1-D_2 (Figura 2).
2. Calcular el valor absoluto del paso anterior $|D_1$-$D_2|$ (Figura 2).
3. Se ordenan de forma ascendente las diferencias absolutas (Figura 3).

GENERACIÓN CON REEMPLAZO				
CARTERAS NO DOMINADAS				
EXPERIMENTO	o3w105	R	Diferencias	Valor Absoluto
1	52	59	-7	7
2	100	86	14	14
3	100	98	2	2
4	50	65	-15	15
5	59	98	-39	39
6	58	81	-23	23
7	94	70	24	24
8	76	90	-14	14
9	51	43	8	8
10	57	80	-23	23
11	68	75	-7	7
12	81	68	13	13
13	60	77	-17	17
14	64	103	-39	39
15	58	74	-16	16
16	56	87	-31	31
17	77	72	5	5
18	82	79	3	3
19	53	102	-49	49
20	74	85	-11	11
21	56	88	-32	32
22	47	88	-41	41
23	86	89	-3	3
24	97	100	-3	3
25	63	66	-3	3
26	65	82	-17	17
27	91	63	28	28
28	93	58	35	35
29	70	56	14	14
30	85	75	10	10

D_1
D_2

Figura 2 Identificación de elementos para prueba Wilcoxon

4. Los valores en cero son descartados (En el ejemplo no hubo valores con 0)

5. Se asigna una categoría creciente (Figura 3).

5.1 Si dos o más valores son iguales se les asigna el promedio de las categorías crecientes (Figura 3).

GENERACIÓN CON REEMPLAZO							
CARTERAS NO DOMINADAS							
EXP	o3w105	R	Diferencias	Valor Absoluto	V.A ordenado	Rango	Promedio de C.C.
1	52	59	-7	7	2	1	1
2	100	86	14	14	3	2	3.5
3	100	98	2	2	3	3	
4	50	65	-15	15	3	4	
5	59	98	-39	39	3	5	
6	58	81	-23	23	5	6	6
7	94	70	24	24	7	7	7.5
8	76	90	-14	14	7	8	
9	51	43	8	8	8	9	9
10	57	80	-23	23	10	10	10
11	68	75	-7	7	11	11	11
12	81	68	13	13	13	12	12
13	60	77	-17	17	14	13	14
14	64	103	-39	39	14	14	
15	58	74	-16	16	14	15	
16	56	87	-31	31	15	16	16
17	77	72	5	5	16	17	17
18	82	79	3	3	17	18	18.5
19	53	102	-49	49	17	19	
20	74	85	-11	11	23	20	20.5
21	56	88	-32	32	23	21	
22	47	88	-41	41	24	22	22
23	86	89	-3	3	28	23	23
24	97	100	-3	3	31	24	24
25	63	66	-3	3	32	25	25
26	65	82	-17	17	35	26	26
27	91	63	28	28	39	27	27.5
28	93	58	35	35	39	28	
29	70	56	14	14	41	29	29
30	85	75	10	10	49	30	30

Figura 3. Asignación de rangos

6. Se asigna el signo de las diferencias en los rangos correspondientes, se suma por separado las diferencias positivas T (+) y negativas T (-). La menor de estas sumas es el valor del estadístico T de Wilcoxon (Figura 4).

GENERACIÓN CON REEMPLAZO							
CARTERAS NO DOMINADAS							
EXP	**o3w105**	**R**	**Diferencias**	**Valor Absoluto**	**Rango**	**T+**	**T-**
1	52	59	-7	7	7.5		7.5
2	100	86	14	14	14	14	
3	100	98	2	2	1	1	
4	50	65	-15	15	16		16
5	59	98	-39	39	27.5		27.5
6	58	81	-23	23	20.5		20.5
7	94	70	24	24	22	22	
8	76	90	-14	14	14		14
9	51	43	8	8	9	9	
10	57	80	-23	23	20.5		20.5
11	68	75	-7	7	7.5		7.5
12	81	68	13	13	12	12	
13	60	77	-17	17	18.5		18.5
14	64	103	-39	39	27.5		27.5
15	58	74	-16	16	17		17
16	56	87	-31	31	24		24
17	77	72	5	5	6	6	
18	82	79	3	3	3.5	3.5	
19	53	102	-49	49	30		30
20	74	85	-11	11	11		11
21	56	88	-32	32	25		25
22	47	88	-41	41	29		29
23	86	89	-3	3	3.5		3.5
24	97	100	-3	3	3.5		3.5
25	63	66	-3	3	3.5		3.5
26	65	82	-17	17	18.5		18.5
27	91	63	28	28	23	23	
28	93	58	35	26	26	26	
29	70	56	14	14	14	14	
30	85	75	10	10	10	10	
					Suma:	**140.5**	**324.5**

Figura 4. Suma de diferencias positivas y negativas

7. Se obtiene los puntos de corte de la distribución a partir de la tabla de Wilcoxon para un nivel de significancia dado. En este ejemplo α=0.05 y *n*= 30

H_0: $\mu = 0$ (No existe diferencia en la cantidad de carteras no dominadas)

H_1: $\mu \neq 0$ (Existe diferencias significativas en la cantidad de carteras no dominadas)

Si p>0.05 Se acepta la hipótesis nula

Si p<= 0.05 Se rechaza hipótesis nula

Aproximación por la normal

T=*min* [T (+), T (-)] ∴ T= *min* [140.5, 324.5]= 140.5

$$Z = \frac{T - \frac{N(N+1)}{4}}{\sqrt{\frac{N(N+1)(2N+1)}{24}}} = \frac{140.5 - \frac{30\ (31)}{4}}{\sqrt{\frac{30(31)(61)}{24}}} = \frac{-92}{48.61841} = -1.892287$$

8. Se rechaza la hipótesis nula si T es menor o igual al valor de la tabla de Wilcoxon (Anexo A)

p= 0.268 *2 (Debido a que es bilateral)

p > 0.058 ∴ 0.0588>0.058

2.9 Conceptos fundamentales de la matemática de intervalos

La concepción de los números de intervalos podría remontarse a Young (1931). Moore y otros eruditos hicieron algunos estudios adicionales sobre números de intervalos (Moore, 1979; Ishihuchi y Tanaka, 1990). Un número de intervalo puede ser visto como una extensión del concepto de un número real como un subconjunto de la recta numérica $\mathbb{R}$ (Moore, 1979).

Moore (1979) describe un número de intervalos en término de un rango como $X = [\underline{X}, \overline{X}]$, dónde $\underline{X}$ denota el límite inferior y $\overline{X}$ el límite superior. Enseguida, se definen las operaciones aritméticas básicas entre intervalos.

El punto clave de los operadores aritméticos básicos, es calcular con conjuntos, cuando sumamos dos intervalos, el intervalo resultante es un conjunto que contiene las sumas de todos los pares de números, uno de cada uno de los conjuntos iniciales. Considerando dos números de intervalos X= $[\underline{X}, \overline{X}]$ e Y= $[\underline{Y}, \overline{Y}]$, las operaciones son definidas de la siguiente manera (Ecuación 1.7 a 1.10, Yamaguchi, 2006).

$$\boldsymbol{X} + \boldsymbol{Y} = [\underline{X} + \underline{Y}, \overline{X} + \overline{Y}] \quad (1.7)$$

$$\boldsymbol{X} - \boldsymbol{Y} = [\underline{X} - \overline{Y}, \overline{X} - \underline{Y}] \quad (1.8)$$

$$\boldsymbol{X} \cdot \boldsymbol{Y} = [\min\{\underline{XY}, \underline{X}\overline{Y}, \overline{X}\underline{Y}, \overline{XY}\}, \ \max\{\underline{XY}, \underline{X}\overline{Y}, \overline{X}\underline{Y}, \overline{XY}\}] \quad (1.9)$$

$$\boldsymbol{X}/\boldsymbol{Y} = [\underline{X}, \overline{X}] \cdot [1/\underline{Y}, 1/\overline{Y}] \quad (1.10)$$

CAPITULO 3

Estado del arte del problema de investigación

El capítulo inicia con la descripción del problema de investigación de la presente tesis, a partir de enunciarlo como un problema de optimización de cartera de proyectos con intervalos y muchos objetivos. Finalmente, para mostrar la relevancia científica del problema estudiado, se describe la complejidad computacional, y se ubica como un problema complejo que es al menos tan difícil como los que están en la clase NP- duro.

3.1 Problema de investigación

En esta tesis se aborda el problema de selección de cartera de proyectos con intervalos y muchos objetivos, en los siguientes puntos, se describe en forma breve que datos de las instancias son los que podrán recibir información en intervalos al tiempo de pasarlo al algoritmo solucionador.

De acuerdo con la literatura el problema de selección de cartera de proyectos (Lin & Shouyang, 2001) son NP- Duros, por lo que al añadir intervalos en la información y ser multiobjetivo es al menos tan complejo como los problemas de esta clase.

3.2 Complejidad del problema de investigación

Se debe demostrar que no existe algoritmo determinista alguno que brinde solución en tiempo polinomial a un problema o que aquellos algoritmos deterministas que lo resuelvan en tiempo no polinomial para definir un problema como intratable.

La teoría de la NP-Completez brinda diversos métodos para probar que un problema es tan complejo que encaja en un selecto grupo de ellos reconocidos por su complejidad al grado que han ofuscado a algunos científicos por años, a estos problemas se les cataloga como problemas NP-Completos (NPC).

Debido a factores comunes que existen en los problemas considerados como NPC, uno de los métodos para demostrar que dos problemas están relacionados es "reducir" uno al otro, para ello se lleva a cabo una transformación que hace corresponder una instancia del primer problema en una instancia equivalente del segundo, por tanto, debido a que existe un algoritmo que brinda una solución para el primer problema en tiempo polinomial, se deduce que ese algoritmo puede transformarse para dar solución al segundo problema.

Para demostrar que un problema pertenece a la clase NPC se deben realizar las siguientes acciones:

1. Demostrar que el problema (P) pertenece a la clase NP.

    a. Construir un algoritmo determinista que verifique que se puede obtener una solución en tiempo polinomial.

2. Validar que C pertenece a la clase NPC.

    a. Obtener una transformación polinomial entre el problema P y un problema del cual se sabe pertenece a la clase NPC y viceversa.

Definición del problema de Selección de Cartera de Proyectos

Dados los proyectos $p_1, p_2, \ldots, p_n$ una cartera de proyectos es un subconjunto de k proyectos ($1 \leq k \leq n$) que cumplen con una serie de restricciones presupuestales.

Problema de Selección de Cartera de Proyectos

Entrada: Un conjunto de proyectos $p_1, p_2, \ldots, p_n$, con sus respectivos costos y beneficios, y una restricción presupuestal *T*.

Pregunta: ¿Existe una Cartera de Proyectos que cumpla con la restricción de presupuesto?

Representación interna del problema.

En el algoritmo se utilizan dos estructuras de datos: una para almacenar los proyectos dados, sus características (costo y beneficio por objetivo) y otra estructura para almacenar las restricciones.

Sea *n* el número de proyectos totales.

Para almacenar cada solución se utiliza una matriz de datos binarios *X* cada fila de esta matriz representa una cartera de proyectos x_i, en donde se almacena un 1 en caso de que el proyecto esté incluido en la cartera o un 0 en caso contrario.

Las restricciones se almacenan en una estructura independiente y, la cual se utilizará únicamente para comprobar que una cartera de proyectos es factible.

Teorema: El problema de Selección de Cartera de Proyectos es NP-Completo.

Demostración:

1. Definir una estructura de dato para representar las soluciones candidatas.

$$x[i] \quad en\ donde: i = 1,2,3, \dots, n$$

2. Construir un algoritmo aleatorio de complejidad *O(n)* que genere una solución candidata.

FOR *i*= 1 TO *n* DO

$$x[i] \coloneqq genera_{aleatoriamente_{componente}(i)}$$

END FOR

3. Construir un algoritmo determinista de complejidad *O(n)* para verificar que cada solución candidata cumple con las especificaciones en el problema.

```
//Verifica que una solución candidata es una cartera de tamaño n.
FOR i = 0 TO n DO
    IF(x[i] == 1) {
     Presupuesto += proyectos [i][0];
     }
```

//Verifica que se encuentre dentro de los límites presupuestales totales

IF (*presupuesto* > *presupuesto_total*)

RETURN false;

RETURN true; /*La cartera *x* es factible*

Con lo anterior se comprueba que el problema de Selección de Cartera de Proyectos pertenece a la categoría NP, el siguiente paso es validar que es NPC. Para comprobarlo se selecciona un problema P' que se sabe es NPC, se genera una función de reducción de P' a P y una función de reducción inversa (P a P'). En el caso del problema de Selección de Cartera de Proyectos se utilizó como P' el problema de Knapsack (Bartlett M., 2005) debido a la similitud en su estructura interna.

Knapsack $<_p$ Selección de Cartera de Proyectos

Descripción de la transformación

Sea *M* un vector de decisión factible en el espacio de búsqueda (mochila) el cual almacena un subconjunto de elementos *P* (paquetes), cada elemento en *P* posee un beneficio (*b*) y un peso (*w*) asociado, además se tiene un peso máximo como restricción asociado al problema el cual indica el rango de pesos en los que *M* es factible. El objetivo es encontrar el subconjunto de elementos en *P* que maximicen el total de beneficios (ecuación 1.11) sin exceder la capacidad de carga C (ecuación 1.12).

$$\max(Q_1, Q_2, \dots, Q_n) \tag{1.11}$$

Sujeta a:

$$\sum_{i=1}^{k} w_i m_i \leq C \tag{1.12}$$

$$Q_i = \sum_{j=1}^{k} b_{i,j} m_j \tag{1.13}$$

Q_i en la ecuación 1.11 representa los objetivos a maximizar, cada uno de los cuales representa el beneficio total de una mochila distinta. w_i en la ecuación 1.12 es el peso asociado al paquete *i*, por otro lado m_i representa el elemento *i* en *M* el cual tendrá el valor de uno en caso de haber sido introducido a la mochila, o cero en caso contrario, esta restricción nos indica que la sumatoria de todos los pesos de los paquetes incluidos en cada mochila no deberá superar el costo máximo *C*, de lo contrario dicha solución será infactible. Mientras que en la ecuación 1.13 $b_{i,j}$ es el beneficio aportado por el elemento *j* a la mochila *i*.

En el caso del problema de Selección de Cartera de Proyectos, sea *X* un vector de decisión factible que representa la cartera de proyectos en el espacio de búsqueda el cual almacena un subconjunto de proyectos *Y*, cada proyecto en *Y* posee un beneficio (*z*) y un costo asociado (*s*), además se tiene un presupuesto total asociado como restricción del problema el cual indica el rango presupuestal con el cual se contrasta la suma de los costos de los proyectos incluidos en una cartera *X* para validar su factibilidad. La meta es encontrar el subconjunto de proyectos en *Y* que maximicen los beneficios de los objetivos (ecuación 1.14) sin exceder el presupuesto total *T* (ecuación 1.15).

Suponiendo que se tiene un vector *X* que cumple las restricciones presupuestales del problema de Selección de Cartera de Proyectos. Para transformar este problema al problema de la mochila.

$$\max (O_1, O_2, \dots, O_n) \tag{1.14}$$

Sujeto a:

$$\sum_{i=1}^{k} s_i x_i \leq T \tag{1.15}$$

En donde:

$$O_i = \sum_{j=1}^{k} z_{i,j} x_j \tag{1.16}$$

$O_i$en la ecuación 1.14 los objetivos a maximizar, S_i en la ecuación 1.15 el costo asociado al proyecto *i*, x_i representa el proyecto *i* en *X* que tendrá el valor de uno en caso de haber sido incluido en la cartera, o cero en caso contrario. Mientras que en la ecuación 1.16 $z_{i,j}$ es el beneficio aportado por el proyecto *j* a la cartera *i*.

Se prueba primero que, si existe X, entonces se tiene una solución factible que satisface los requerimientos del problema de Knapsack.

Suponiendo que existe una *M*, esto significa que al menos hay una solución factible la cual genera el conjunto *Q*, entonces debido a la similitud existente entre las ecuaciones que definen cada uno de estos dos problemas, existe también una cartera *X* que da solución al problema de Selección de Cartera de Proyectos.

Se prueba ahora que, si existe una cartera X, también hay una M que hace lo propio con el problema de Knapsack.

Sea *X* una cartera factible que soluciona el problema de Selección de Cartera de Proyectos, entonces como ya se mencionó, debido a la similitud y paralelismo entre estos dos problemas debe existir una *M* que satisface de la misma forma al problema de Knapsack.

Por tanto, el problema de Selección de Cartera de Proyectos es NP-Completo.

3.3 Estado del arte

Los primeros autores en aplicar matemáticas de intervalos para modelar el conocimiento imperfecto en la optimización de cartera de proyectos fueron (Liesio, 2007). Otros enfoques basados en intervalos fueron propuestos por (Fliedner & Liesio, 2016, Toppila & Salo, 2017, Liesio et al., 2008, Balderas et al., 2016). Sin embargo, existen algunos problemas abiertos en esta área de investigación; por ejemplo, algunos enfoques utilizan modelos muy simples de preferencias del DM; otros no permiten ajustar un nivel de conservadurismo asociado con la subjetividad del DM o manejan funciones con pocos objetivos.

Preference programming for robust portfolio modeling and project selection.

Liesio, 2007

Desarrolla un enfoque RPM (Robust Portfolio Modeling), en donde la información imperfecta impacta los criterios de los pesos e ingresos de los proyectos, mediante el uso de una función de pesos de suma ponderada, y define el concepto de dominancia entre carteras. La definición de dominancia basada en una función de pesos de suma ponderada es una manera de incorporar las preferencias del DM.

Este enfoque identifica soluciones para las cuales ninguna cartera factible produce mayor valor para todas las posibles realizaciones inciertas de los puntajes del proyecto y criterios de pesos. Al requerir un mayor valor para todas las posibles realizaciones de los parámetros inciertos, RPM es probablemente un enfoque muy conservador.

Robust portfolio modeling with incomplete cost information and project interdependencies

Liesio, 2008

Este trabajo representa una extensión al trabajo de RPM (Liesio, 2007) que incluye sinergia, información de costos incompleta y niveles de presupuesto variables presente en las carteras de proyectos multiobjetivo el cual conduce a un problema de programación lineal con coeficientes de función objetivo con valores de intervalo.

Adjustable robustness for multi-attribute project portfolio selection

Fliedner and Liesio, 2016

Inspirado en RPM, Fliedner y Liesio introducen un método que permite obtener soluciones relacionadas a diferentes niveles de conservadurismo por parte del DM, permitiendo de una manera flexible ajustarlas. Dado que una función de pesos de suma ponderada requiere propiedades especiales (ejemplo: comparabilidad, preferencias transitivas, intercambios

constantes de compensación) y una articulación “a priori” de las preferencias del DM, un enfoque “a posteriori” para la incorporación de preferencias pudiera ser una mejora.

Binary decision diagrams for generating and storing non-dominated project portfolios with interval-valued project scores

Toppila and Salo, 2016

Proponen un método basado en la definición de dominancia usando intervalos y un algoritmo eficiente para encontrar toda la frontera de Pareto de soluciones no dominadas, de las cuales el DM debe seleccionar el mejor compromiso. Este enfoque supera las limitaciones de la función de pesos de suma ponderada; sin embargo, la definición de dominancia es conservadora y no se pueden explorar diferentes niveles de conservadurismo de parte del DM; además, el método solamente es operacional en problemas con pocas funciones objetivo; si el número de funciones objetivo incrementa, las limitaciones cognitivas del DM hacen muy difícil la selección final.

Metaheuristic robust optimization of Project portfolios using an interval-based model of imprecisions

Balderas, 2016

El trabajo de Balderas propone un algoritmo genético, basado en el NSGA-II. Proponen una medida de calidad basada en el grado paretiano, sin embargo, únicamente maneja los niveles de conservadurismo en el presupuesto disponible y no incorpora un método de incorporación de preferencias, lo hacen utilizando TOPSIS-Gris que fue propuesto por (Lin, 2008).

Modelando la imprecisión del problema de cartera de proyectos con filosofía gris.

Balderas, 2018

Este trabajo desarrolla el tratamiento de conocimiento imperfecto inmerso en el problema de cartera de proyectos utilizando matemáticas de intervalos empleando un algoritmo

evolutivo e incorporación de preferencias del decisor en el proceso de optimización con la finalidad de dirigir la búsqueda hacia la región de interés en lugar de buscar el frente completo.

Trabajos	Algoritmo	Objetivos	Uso de matemáticas de intervalos
Lieso, 2007	Programación dinámica	4	Si
Lieso, 2008	Programación lineal	3 y 5	Si
Fliedner & Lieso, 2016	Programación dinámica	3	Si
Balderas, 2016	A.G.	2	Si
Toppila & Salo, 2016	Diagrama de decisión binaria	4	Si
Balderas, 2018	A.G.	9	Si
Esta tesis	A.G.	2,3,4,8,9,13 y 15	Si

Tabla 1. *Comparación de trabajos del estado del arte contra la propuesta de investigación de este trabajo.*

CAPÍTULO 4

Metodología de solución

La selección de cartera de proyectos multiobjetivo con intervalos (UPPS) es el objeto de estudio de esta tesis. En esta sección, se describe la metodología propuesta para dar solución al problema de cartera de proyectos multiobjetivo. El método propuesto es un algoritmo evolutivo multiobjetivo conocido como MOEA/D (Multiobjective Evolutionary Algorithm Based on Decomposition) una propuesta de Zhang y Li en (Zhang y Li, 2007).

4.1 Adaptación de intervalos a la Selección de cartera de proyectos multiobjetivo

Suponga una cartera de N proyectos $\overrightarrow{x_i} = \langle x_1, x_2, \dots, x_n \rangle$

$$\text{Donde } \overrightarrow{x_i} = \begin{cases} 1 \; si \; el \; proyecto \; i \; pertenece \; a \; la \; cartera \\ 0 \; si \; el \; proyecto \; i \; no \; pertenece \; a \; la \; cartera \end{cases}$$

n= Número de proyectos

$[\underline{B}, \overline{B}]$= Límite inferior y superior del presupuesto de la cartera.

$[\underline{c}, \overline{c}] = [\underline{c}, \overline{c}]_1, [\underline{c}, \overline{c}]_2, \dots, [\underline{c}, \overline{c}]_n$ donde $[\underline{c}, \overline{c}]_i \in n$ representa el límite inferior y superior del costo de seleccionar el proyecto i en la cartera

Asimismo, un par de vectores $a = (a_{11}, a_{12}, \dots, a_{1m}, a_{21}, \dots, a_{nm})$ y $r = (r_{11}, r_{12}, \dots, r_{1l}, r_{21}, \dots, r_{nl})$donde $a_{ij} \in \{0,1\}$ representa si el proyecto i pertenece al área j y $r_{ik} \in \{0,1\}$ representa si el proyecto i pertenece a la región k.

Entonces el problema se resuelve al encontrar una cartera $\vec{x}$ que satisfaga las siguientes restricciones.

$$\underline{B} \leq \sum_{i=1}^{N} x_i c_i \leq \overline{B}$$

$$[\underline{A},\overline{A}]_1^L \le \sum_{i=1}^{N} x_i[\underline{c},\overline{c}]_i a_{i1} \le [\underline{A},\overline{A}]_1^U \qquad [\underline{R},\overline{R}]_1^L \le \sum_{i=1}^{N} x_i[\underline{c},\overline{c}]_i r_{i1} \le [\underline{R},\overline{R}]_1^U$$

$$[\underline{A},\overline{A}]_2^L \le \sum_{i=1}^{N} x_i[\underline{c},\overline{c}]_i a_{i2} \le [\underline{A},\overline{A}]_2^U \qquad [\underline{R},\overline{R}]_2^L \le \sum_{i=1}^{N} x_i[\underline{c},\overline{c}]_i r_{i2} \le [\underline{R},\overline{R}]_2^U$$

$$[\underline{A},\overline{A}]_m^L \le \sum_{i=1}^{N} x_i[\underline{c},\overline{c}]_i a_{im} \le [\underline{A},\overline{A}]_m^U \qquad [\underline{R},\overline{R}]_k^L \le \sum_{i=1}^{N} x_i[\underline{c},\overline{c}]_i r_{ik} \le [\underline{R},\overline{R}]_k^U$$

$[\underline{A},\overline{A}]_j^U$ =Limite inferior y superior del límite superior de inversión en área j

$[\underline{A},\overline{A}]_j^L$ =Límite inferior y superior del límite inferior de inversión en área j

$[\underline{R},\overline{R}]_j^U$ =Límite inferior y superior del límite superior de inversión en región j

$[\underline{R},\overline{R}]_j^L$ =Límite inferior y superior del límite inferior de inversión en región j

Asimismo, incluye el vector $[\underline{b},\overline{b}] = \left([\underline{b},\overline{b}]_{11}, [\underline{b},\overline{b}]_{12}, \dots, [\underline{b},\overline{b}]_{1o}, [\underline{b},\overline{b}]_{21}, \dots, [\underline{b},\overline{b}]_{no}\right)$ donde b_{ij} indica el beneficio aportado en el objetivo j por el proyecto i. Entonces la función objetivo se define como:

$$f(\vec{x}) = \{f_1(\vec{x}), f_2(\vec{x}), \dots, f_p(\vec{x})\}$$

Donde:

$$f_i(\vec{x}) = \sum_{j=1}^{N} x_j[\underline{b},\overline{b}]_{ji}$$

La mejor solución compromiso se define al resolver

$$\max_{x \in S_F}\left(f(\vec{x})\right)$$

Donde $S_F$es el espacio de soluciones factibles.

4.2 Generador de instancias de pesos

Este método se basa en las dos propiedades de la función logaritmo natural: $ln(1) = 0$ y $ln(e) = 1$ donde los parámetros de GW (ecuación 1.17) se ajustan a cambiar suavemente los pesos en [0,1]. Sea $\lambda = (\lambda_1, \lambda_2, \dots, \lambda_m)$ el vector de pesos tal que $0 \leq \lambda_k \leq 1$ para todo $k = 1,2, \dots, m$ y $\sum_{k=1}^{m} \lambda_k = 1$ (Wu, Kwong, Yuheng, Ke, & Qingfu, 2017) .

$$\lambda_1(g) = ln\left[\frac{4 * i * e}{FQ}\right] + cos\left[\frac{2 * \pi * i}{FQ}\right] \tag{1.17}$$

$$\lambda_{j-1}(g) = \left(1 - \lambda_j\right) * ln\left[\frac{4 * j * e}{FQ}\right] + cos\left[\frac{2 * \pi * j}{FQ}\right] \tag{1.18}$$

$$\lambda_m(g) = 1.0 - \sum_{j=2}^{m} \lambda_{j+1} \tag{1.19}$$

Donde:

$g =$ Es el índice de generación

$e = exp$ (1)

$FQ =$ Frecuencia de cambio de peso

$m =$ Número de objetivos

En el algoritmo 4 se muestra el proceso para la generación de pesos donde la frecuencia de cambio es un número entero, además FQ permite establecer el número máximo de vectores de peso que se generaran. Por ejemplo, si FQ es establecido en 800 se generan 200 vectores de peso $V = \frac{FQ_{m-1}}{4}$ con *m*=3 números de objetivos. Para la columna 1, se realiza la línea 4, en donde el valor de λ se encuentra entre 0 y 1, para las columnas intermedia se realiza una condición (línea 5) donde *j* es la posición de la columna en la que actualmente está posicionado el puntero, donde *j* sea mayor que 0, es decir, que no sea la posición del objetivo 1, pero que además no sea la última columna (*j* < *m-1*), así que, sí la posición

actual es el objetivo 2 de 3, entraría en esta condición. Dentro de esta sentencia se genera λ teniendo una diferencia de (1 - la sumatoria de los λ anteriores) multiplicado por $\ln\left[\left(\frac{4*j*e}{FQ}\right)+\cos\left(\frac{2*\pi*j}{FQ}\right)\right]$ como se muestra en la línea 6. Por último, en la columna del objetivo *m* solo se realiza la diferencia de (1 - la sumatoria de los λ anteriores) como en la línea 8. Para comprobar que es un peso aceptable, el resultado de la suma de los *m* objetivos debe ser 1. La línea 9 es una variable utilizada para sumar los λ por fila y cuando termina los *m* objetivos, la variable *aux* es inicializada en 0 (Línea 11).

Algoritmo 4. Algoritmo GW

Entrada:
m: Número de objetivos
FQ: Frecuencia de cambio
Salida:
$\lambda =$ vector de pesos
1. **Para** *i* **desde** 0 hasta $\frac{FQ}{4}$ **hacer**:
2. **Para** *j* **desde** 0 hasta *m* **hacer**
3. **Si** *j* = 0 **hacer**
4. $\lambda = \ln\left[\left(\frac{4*i*e}{FQ}\right)+\cos\left(\frac{2*\pi*i}{FQ}\right)\right]$
5. **Si** *j* > 0 **&** *j* < (*m* - 1) **hacer**
6. $\lambda = (1.0 - aux) * \ln\left[\left(\frac{4*j*e}{FQ}\right)+\cos\left(\frac{2*\pi*j}{FQ}\right)\right]$
7. **Si** *j* = (*m*-1) **hacer**
8. $\lambda = 1.0 - aux$
9. *aux* += λ
10. **Fin para**
11. *aux* = 0
12. **Fin para**

4.3 Generador de instancias de cartera de proyectos multiobjetivo con intervalos

El algoritmo 5 muestra el pseudocódigo del generador de instancias propuesto para PPS con intervalos. Los parámetros configurables por el usuario para crear una instancia son: presupuesto, número de objetivos, proyectos, áreas y regiones, y límites de costes y objetivos. Los resultados son los valores de intervalo que definen el presupuesto, las áreas, las regiones y para los proyectos sus costes, valores de objetivos y el área y la región a la que pertenecen.

El generador crea un presupuesto de intervalo en la Línea 0 en función del presupuesto de entrada B. La Figura 5 muestra un ejemplo de la definición de tales intervalos.

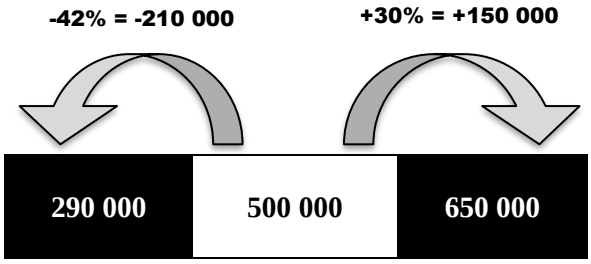

Figura 5. Parámetro de presupuesto

De la línea 1a la 6, el generador crea los valores para las áreas de la instancia. De tal manera que utiliza el presupuesto *B* para definir los límites adecuados a los valores de las áreas (líneas 1 y 2). Después de eso, eligió aleatoria-mente los valores con esos límites como los valores delimitadores de cada área (Líneas 4 y 5). De las líneas 7 a 12, el generador asigna valores a las regiones de una manera similar a la realizada en las áreas; es decir, utiliza el presupuesto para definir los límites máximos adecuados, y con ellos eligió aleatoriamente valores para enlazar las regiones distintas.

El siguiente paso en el generador es la definición de valores para los proyectos. Las líneas 14 a 26 realizan esta tarea. En primer lugar, el área y las regiones se eligen aleatoriamente en las líneas 14 y 15. Después, el costo del proyecto se crea dentro de los límites proporcionados como entradas (Líneas 16 a 17). De las líneas 18 al 26 el proceso genera los valores para los objetivos de un proyecto. Utiliza dos estrategias, una basada en los costos del proyecto (Línea 20), y la otra basada en los límites de los objetivos establecidos como insumos (Línea 22). Con el valor o el generador crea un intervalo para el objetivo utilizando el 80% de él como límite inferior y 120% como límite superior.

Algoritmo 5. Generador de instancias PPS con intervalos

Entrada:

- B← Presupuesto (sin intervalo)
- $\{m, p, a, r\}$← Numero de objetivos, proyectos, áreas y regiones
- $\left[\underline{c}, \overline{c}\right]$← Límites extremos de costes del proyecto (sin intervalos)
- $\left[\underline{m}, \overline{m}\right]$← Objetivos límites extremos

Salida:

- $\left[\underline{B}, \overline{B}\right]$← Presupuesto como intervalo
- $\left\{\left[\underline{a_1}, \overline{a_1}\right], \left[\underline{a_2}, \overline{a_2}\right], \ldots, \left[\underline{a_a}, \overline{a_a}\right]\right\}$← Límites de cada área *i*
- $\left\{\left[\underline{r_1}, \overline{r_1}\right], \left[\underline{r_2}, \overline{r_2}\right], \ldots, \left[\underline{r_r}, \overline{r_r}\right]\right\}$← Límites de cada región *r*
- $\{\{C_1, A_1, R_1\}, \ldots, \{C_p, A_p, R_p\}\}$← Costo, Área y región para cada proyecto *p*
- $\left\{\left[\underline{f_{1p}}, \overline{f_{1p}}\right], \left[\underline{f_{2p}}, \overline{f_{2p}}\right], \ldots, \left[\underline{f_{mp}}, \overline{f_{mp}}\right]\right\}$←Beneficio de los objetivos *m* de cada proyecto p (en intervalos)

0. $\left[\underline{B}, \overline{B}\right] = [0.58B, 1.3B]$
1. $\left[\underline{a_l}, \overline{a_l}\right] = [(0.7 * B)/(1.7a+0.1a^2), (1.27 * B)/(1.7a+0.1a^2)]$
2. $\left[\underline{a_u}, \overline{a_u}\right] = [((2.159 +0.127a)*B) / a, ((2.635+0.155a) * B)/a]$
3. **for each** $i \in \{1, 2, \ldots, a\}$ **do**
4. $\underline{a_i} = \underline{a_l} + \text{Random}(\overline{a_l} - \underline{a_l})$
5. $\overline{a_i} = \underline{a_u} + \text{Random}(\overline{a_u} - \underline{a_u})$
6. **end**
7. $\left[\underline{r_l}, \overline{r_l}\right] = [(0.8 * B)/(1.7r+0.1r^2), (1.2 * B)/(1.7r+0.1r^2)]$
8. $\left[\underline{r_u}, \overline{r_u}\right] = [((1.02+0.06r)*B)/r, ((2.38 +0.14r)*B)/r]$
9. **for each** $i \in \{1, 2, \ldots, r\}$ **do**
10. $\underline{r_i} = \underline{r_l} + \text{Random}(\overline{r_l} - \underline{r_l})$
11. $\overline{r_i} = \underline{r_u} + \text{Random}(\overline{r_u} - \underline{r_u})$
12. **end**
13. **for each** $i \in \{1, 2, \ldots, p\}$ **do**
14. A_i= Random(a)
15. R_i= Random(r)
16. $v = \underline{c} + \text{Random}(\overline{c} - \underline{c})$
17. $\left[\underline{C_i}, \overline{C_i}\right] = [0.99 * v, 1.2 * v]$
18. **for each** $j \in \{1,2, \ldots, m\}$ **do**
19. **if (**Random.nextBoolean()**) then**
20. obj = Random(($v - \underline{c}$)/ ($\overline{c} - \underline{c}$))
21. **else**
22. $o = \underline{m} + \text{Random}(\overline{m} - \underline{m})$
23. **end**
24. $\underline{f_{ij}}$= 0.8*o
25 $\overline{f_{ij}}$ = 1.1*o
26. **end**
27.**end**

4.4 Propuesta de solución

Para iniciar el proceso se necesitan instancias específicas del problema de cartera de proyectos o en otro caso, generar instancias con intervalos en ciertos datos. Después, la instancia se incorpora al algoritmo MOEA/D (Zhang y Li, 2007) que se ha modificado para la lectura de información con intervalos, en la Figura 6, se observa en rojo las partes del algoritmo MOEA/D que se han modificado.

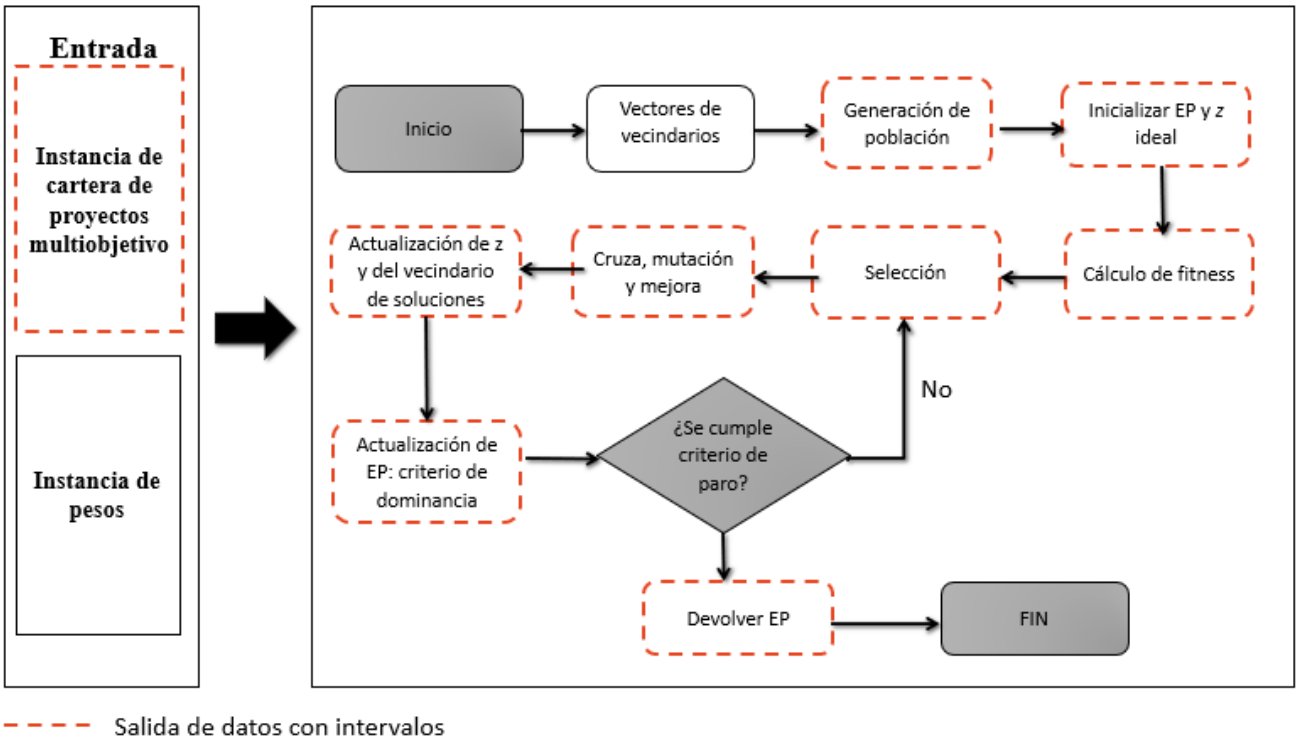

Figura 6. Diseño de la propuesta

Generar población inicial

Esta sección describe la estrategia de inicialización para la población de I-MOEA / D, en presencia de intervalos. El proceso es sencillo; elige un proyecto i como parte de la cartera siempre que un valor uniforme aleatorio v se encuentre por debajo de una selección de umbral de selección predefinida, β (establecido en 0.5 para este trabajo de investigación).

Siguiendo un enfoque de prueba y error, el algoritmo descarta aquellas soluciones que se volvieron inviables en el proceso. El algoritmo 6 muestra el pseudocódigo del método.

Algoritmo 6. Generación de población inicial

Input:

- β-Umbral de selección
- *m*-Número de objetivos
- *p*-Total de proyectos
- *a*-Numero de áreas
- *r*-Numero de regiones

Output:

- Población inicial

```
0.  x⃗={1,1,…,1}
1.  while (!Factibilidad (x⃗)) do
2.    x⃗={0,0,…,0}
3.    for each i ∈ {1,2, … , p} do
4.      r=random (0,1)
5.      If (r < β) then
6.        x_i = 1
7.        c(x⃗) += c(i)
8.        for each j ∈ {1, … , m} do
9.          f_i(x⃗) += f_j(i)
10.         end
11.         for each j ∈ {1, … , a} do
12.           A_i(x⃗) += A_j(i)
13.         end
14.         for each j ∈ {1, … , r} do
15.           R_i(x⃗) += R_j(i)
16.         end
17.       end
18.     end
19.  end
20. return x⃗
```

La línea 1 usa la función *Factibilidad* para asegurar una solución factible; valida las restricciones de las UPPS de presupuesto, área y región. El algoritmo prueba cada proyecto para ser incluido en la cartera en las líneas 4 y 5. Siempre que se cumple la condición, se acumulan los costos y valores para los objetivos, áreas y regiones (líneas 7, 8, 11 y 14, respectivamente).

La viabilidad requiere operaciones adicionales y relacionales. Dados dos números de intervalo $\boldsymbol{E} = [\underline{E}, \overline{E}]$ y $\boldsymbol{D} = [\underline{D}, \overline{D}]$, el resultado de **C** = **E** + **D** se puede calcular como $\boldsymbol{C} = [\underline{E} + \underline{D}, \overline{E} + \overline{D}]$. Por otro lado, la operación relacional $\boldsymbol{D} \leq \boldsymbol{E}$ se puede estimar usando el cociente relacional definido por la ecuación 1.20.

$$p_{ED} = \frac{\overline{E} - \underline{D}}{\left(\overline{E} - \underline{E}\right) + \left(\overline{D} - \underline{D}\right)} \tag{1.20}$$

Con base en el cociente relacional, la ecuación 1.21 define la medida de posibilidad de $Poss(\boldsymbol{D} \leq \boldsymbol{E})$ utilizada para expresar la relación deseada entre los intervalos D y E. Este trabajo establece que $Poss(\boldsymbol{D} \leq \boldsymbol{E}) \geq 0.5$.

$$Poss(\boldsymbol{D} \leq \boldsymbol{E}) = \begin{cases} 1 & \text{if } p_{ED} > 1, \\ p_{ED} & \text{if } 0 \leq p_{ED} \leq 1, \\ 0 & \text{if } p_{ED} \leq 0 \end{cases} \tag{1.21}$$

La Figura 7 muestra el proceso realizado por el Algoritmo 1 en la construcción de una cartera. Esta figura muestra una matriz con 25 celdas que representan una cartera y los 25 proyectos potenciales. Cada celda también representa una iteración y su valor generado aleatoriamente. Tenga en cuenta que las celdas de sombra corresponden a aquellas en las que el valor aleatorio se encuentra por debajo del β = 0.5. El proceso se repite tantas soluciones como tenga la población inicial de I-MOEA / D.

1	2	**3**	4	5	**6**	7	8
0.89	0.59	**0.16**	0.76	0.68	**0.42**	0.73	0.96

9	10	**11**	12	**13**	14	15	16
0.09	0.53	**0.38**	0.71	**0.12**	0.83	0.88	0.65

17	18	19	**20**	21	22	23	**24**
0.99	0.79	0.55	**0.36**	0.68	0.62	0.92	**0.28**

25
0.64

Valor aleatorio seleccionado

Figura 7. Representación gráfica del proceso iterativo del algoritmo 6 para construir una cartera.

Mientras que la Figura 7 muestra el proceso de selección de proyectos, la Figura 8 muestra la representación binaria requerida por I-MOEA / D. La cartera debe ser factible y deben satisfacerse todas las limitaciones de costos, áreas y regiones.

1	2	3	4	5	6	7	8
0	0	1	0	0	1	0	0

9	10	11	12	13	14	15	16
1	0	1	0	1	0	0	0

17	18	19	20	21	22	23	24
0	0	0	1	0	0	0	1

25
0

Figura 8. Vector binario que representa una solución (o cartera) utilizada por I-MOEA / D. Aquí el valor 1 significa que el proyecto es parte de la cartera y 0 en caso contrario.

I-MOEA/D

I-MOEA/D es una variante de MOEA/D que resuelve PPS con incertidumbre; implementa operadores evolutivos para manejar intervalos. Los intervalos representan una media de expresión de la incertidumbre en los valores de los objetivos, los costos y los recursos. El algoritmo 7 muestra el pseudocódigo general de la estrategia propuesta. I-MOEA/D da una población externa (*EP*) que contiene las soluciones no dominadas encontradas durante el proceso de optimización.

Algoritmo 7. I-MOEA/D

Input:

- *MOP*= Problema de optimización Multiobjectivo
- *N*= Tamaño de población
- *p*=Numero de proyectos
- *m*= Numero de objetivos
- *T= Tamaño del vecindario de los vectores de peso*
- *EvaluaciónMaxima*= Numero de generaciones

Output:

EP= Población externa

0. *W* = LeerVectorDePesos()
1. *EP*= ϕ
2. ***Calcular_Distancia_Euclidiana* (W)**
3. *OrdenarVector* ()
4. **Población=*GenerarPoblaciónInicial*()**
5. *Inicializar_Z (Población)*
6. Generaciones=0;
7. **While**(Generaciones < *EvaluaciónMaxima*)
8. *i*=1
9. **For each** $i \in N$ **do**
10. [p_1, p_2] =**SelecciónPorTorneo** (Population, *B(i)*, *T*)
11. *descendencia*=CruzaDeUnPunto(*p*, [p_1,p_2])
12. *descendencia* =MutaciónDeGen(*p*, *descendencia*)
13. *descendencia*=**MejoraMutaciónDeGen** (*descendencia*)
14. ***ActualizarZ*** ($\vec{z}$, *f*(*descendencia*))
15. ***ActualizarEP*** (EP, *f*(*descendencia*))
16. **end**
17. Generaciones++
18. **end**

Los vectores binarios $\vec{x} = \langle x_1, x_2, \ldots, x_p \rangle$ codifican una cartera o solución proporcionada por el algoritmo. Tales vectores son cromosomas en el enfoque evolutivo, y los índices de los vectores de la matriz son alelos que denotan proyectos distintos.

Los métodos de I-MOEA/D que se distinguen de las implementaciones de otros MOEA/D son cinco, que aparecen en negrita en el Algoritmo 7 (Líneas 2, 4, 10, 13-15). Estos métodos son la función de inicialización, el operador de selección, el operador de reparación/mejora, y la actualización del vector *Z* y del conjunto *EP*. El resto de la sección proporciona una descripción detallada de los métodos.

En la primera fase, el algoritmo I-MOEA/D crea el conjunto inicial del vector de pesos (línea 0) e inicializa el *EP* en vacío (línea 1). También inicializa los vectores de peso $\{w_1, w_2, \ldots, w_N\}$, calcula la distancia euclidiana entre ellos e inicializa la vecindad de sus vectores *B(i)*, $1 \leq i \leq N$ (Líneas 2 a 4). El Algoritmo 7 rellena el vector de soluciones inicial *B* de tamaño *N* y asocia a cada solución B_i un vector de pesos w_i. Entonces, la vecindad *B*(*i*) de un vector de pesos w_i contiene los vectores de pesos más cercanos indexados por distancia euclidiana. Finalmente, la fase de inicialización rellena el vector *Z*, el vector de los mejores valores objetivo encontrados en el proceso de búsqueda (Línea 5). El vector *B* corresponde a la población inicial. El bucle principal del I-MOEA/D comienza teniendo como criterio de parada un número máximo de evaluaciones previamente definido (Línea 7).

Selección por torneo. El método de selección selecciona de la población dos soluciones al azar. A continuación, las compara por su coste y asigna la mejor como primer padre p1, y la otra como segundo padre p2. El algoritmo 8 devuelve ambos padres (Miller & Goldberg, 1995). Este método requiere comparar el coste utilizando la comparación relacional de intervalos.

Algoritmo 8. Algoritmo de Selección por torneo

Entrada

- *B*=Población
- $B_{i=}$ Vecindad de cada vector de peso *i*
- T= Número de vectores de peso en la vecindad de cada vector

Salida

- Padre p1 y p2

0. **While** (k==l) **do**
1. k = *Aleatorio* ()
2. l = *Aleatorio* ()
3. **end**
4. *x* = $B_{i,k}$
5. *y* = $B_{i,l}$
6. **If** (c (B[x]) < c (B[y]) **then**
7. p_1=B[x]
8. p_2=B[y]
9. **else**
10. p_1=B[y]
11. p_2=B[x]
12. **end**

Primeramente, se eligen aleatoriamente dos de las carteras (Línea 1 y 2), pero se revisa que las dos carteras (*k* y *l*) no sean iguales (Línea 0), si se cumple la condición a *x* se le asigna el valor de $B_{i,k}$ (Línea 4) y a *y* se le asigna $B_{i,l}$ (Línea 5), a p_1 se le asigna el valor de *B[x]* (Línea 7) y a p_2 se le asigna el valor de *B[y]* (Línea 8), si *B[x]< B[y]*. En caso contrario p_1 toma el valor de *B[y]* (Línea 10) y p_2 es igual a *B[x]* (Línea 11).

Cruza de un punto. Los dos padres que fueron elegidos en la selección por torneo combinan sus cromosomas, tomando un punto aleatorio el cual indica el corte para heredar los genes a el nuevo hijo. Esta técnica fue propuesta por Holland en (Holland, 1975), ahora ha sido adaptada para el problema de cartera de proyectos multiobjetivo con intervalos (Algoritmo 9).

Algoritmo 9. Algoritmo de Cruza simple o de un punto

Entrada

- P=Número de proyectos
- [p_1, p_2]= Padres

Salida

- y'= Hijo

0. corte = Aleatorio (1, p-1)
1. y [0, …, corte-1] = p_1[0, …, corte-1]
2. y [corte, …, p-1] = p_2[corte, …, p-1]
3. **Return** y

El algoritmo 9 tiene dos fases. Primero se elige un corte al azar, y debe estar entre 1 y p-1, donde p es el número de proyectos (línea 0). Después, se crea el hijo usando genes de los padres p_1, p_2. El primer progenitor transmitirá los genes correspondientes a los alelos en los índices 0 al corte - 1 de su vector correspondiente (Línea 1); este es el mejor padre de ambos por costo. El segundo padre donará los genes de los índices de su vector desde el corte hasta p - 1 (línea 2). El nuevo hijo es la descendencia que devuelve el método. La Figura 9 muestra una descripción gráfica de cómo los genes de los padres se heredan al niño usando nuestro método.

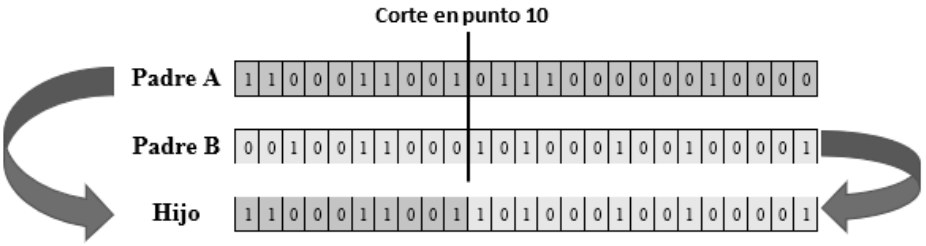

Figura 9. Asignación de conjunto de genes de padres a hijo

Mutación de gen. El operador de mutación elegido es una mutación simple. Este proceso selecciona un alelo de la solución y cambia su valor. Dado que la solución es un vector binario, el alelo elegido cambiará su valor de 1 a 0, o viceversa (Díaz, 2014). El algoritmo 10 muestra esta estrategia.

Algoritmo 10. Algoritmo de Mutación de gen

Entrada

- *p*= Total de proyectos
- *y*= Solución para mutar

Salida

- *y'*= Hijo Mutado

0. *r*= Aleatorio (0, p – 1)
1. *y' = y*
2. *y'[r] = (y'[r] + 1) % 2;*
3. **return** *y'*

Después de aplicar los operadores genéticos por I-MOEA / D (Líneas 10-12, Algoritmo 7), la solución generada pasa a un proceso de reparación / mejora (Línea 13, Algoritmo 7). El método repara / mejora una solución al hacer factibles las soluciones inviables. La estrategia utilizada saca proyectos aleatoriamente hasta el cumplimiento de las restricciones. Este procedimiento requiere la implementación de operaciones de intervalo, tanto aritméticas como relacionales.

Cada iteración de I-MOEA / D actualiza los vectores $\vec{z}$, B (o Población), y EP con la descendencia. La descendencia sustituye los valores de los objetivos ideales en $\vec{z}$ si es necesario. El conjunto de EP debe eliminar las soluciones dominadas por la descendencia e incluirlo si está dominado por nadie. La condición de dominancia usa las operaciones de relación de intervalo definidas previamente.

Finalmente, cuando se cumple el criterio de paro, I-MOEA / D notifica la EP configurada como la región de interés aproximada.

CAPITULO 5

Experimentación y resultados

Esta sección contiene una serie de experimentos destinados a validar la calidad del I-MOEA / D en comparación con el algoritmo I-NSGA-II (Balderas, 2018). La configuración del diseño experimental tuvo en cuenta los siguientes detalles: a) el tamaño del conjunto de instancias UPPS probadas fue de 7; b) el conjunto de proyectos involucrado fue siempre de cardinalidad 100; c) el número de objetivos involucrados en los casos, varió según {2,3,4,8,9,13,15}; d) el generador aleatorio propuesto que se muestra en la sección 2.4 generó las instancias. En cuanto a los algoritmos, el tamaño de la población fue de 100, el criterio de parada fue después de 500 generaciones y los operadores de cruce y mutación consideraron una probabilidad del 100%.

El entorno de prueba del algoritmo se implementó en el lenguaje de programación Java y se ejecutó en una computadora con las siguientes características: CPU Intel Core i5 de 220 GHz, 4 GB de RAM y sistema operativo Windows 10.

El número de carteras no dominadas y la cardinalidad de las carteras son las dos medidas de calidad de interés de este trabajo para evaluar el rendimiento de los algoritmos con fines comparativos. Las ecuaciones (1.22) a (1.25) muestran los indicadores formados a partir de las mediciones anteriores, donde EP es el conjunto final no dominado de soluciones del algoritmo después de 30 ejecuciones independientes. Señalemos que los valores de los indicadores más grandes representan un mejor rendimiento en un algoritmo.

$$I_1 = |EP| \tag{1.22}$$

$$I_2 = \frac{\sum_{x \in EP} |x|}{|EP|} \tag{1.23}$$

$$I_3 = \min_{x \in EP}\{|x|\} \tag{1.24}$$

$$I_4 = \max_{x \in EP}\{|x|\} \tag{1.25}$$

La Tabla 2 compara I-MOEA / D e I-NSGA-II. Las columnas 1 y 2 muestran los nombres de las instancias y los algoritmos, respectivamente. Las columnas 3 a 6 muestran los valores de los indicadores. Tenga en cuenta que el nombre codificado *oipj* contiene el número de objetivos *i* y proyectos *j*.

Según la Tabla 2, I-NSGA-II mejora I-MOEA / D en la instancia con dos objetivos. Las diferencias van del 5% al 12% en los valores observados de los indicadores; esta es una condición común ya que NSGA-II generalmente tiene un buen desempeño en ese número de objetivos.

En consecuencia, I-MOEA/D es el claro ganador en los casos restantes; sus diferencias de rendimiento varían del 69% al 97%. En conclusión, los resultados generales que se muestran en la Tabla 3 demuestran que I-MOEA/D mejora I-NSGA-II en todos los indicadores. Estos resultados también indican un rendimiento deficiente de I-NSGA-II en problemas de muchos objetivos, una condición previamente observada.

Instancia	Algoritmo	I_1	I_2	I_3	I_4
o2p100	I-NSGA-II	63	57	56	58
	I-MOEA/D	55	54	53	55
o3p100	I-NSGA-II	526	39	37	40
	I-MOEA/D	4556	44	40	46
o4p100	I-NSGA-II	139	58	57	59
	I-MOEA/D	449	68	67	68
o8p100	I-NSGA-II	585	38	35	40
	I-MOEA/D	21327	46	41	47
o9p100	I-NSGA-II	579	41	38	42
	I-MOEA/D	27863	45	40	47
o13p100	I-NSGA-II	521	52	49	54
	I-MOEA/D	5123	63	61	64
o15p100	I-NSGA-II	677	34	31	37
	I-MOEA/D	21417	47	41	48

Tabla 2. *Comparación de I-MOEA/D e I-NSGA-II por Indicadores de calidad.*

Para proporcionar más información, calculamos las diferencias relativas entre los indicadores medidos para I-MOEA / D e I-NSGA-II. Para este propósito, la ecuación 1.26 define una métrica para calcular el porcentaje de mejora logrado por el algoritmo ganador para el indicador dado I^k_j, donde j es el indicador y k = 1 si el algoritmo es I-MOEA / D o k= 2 si es I-NSGA-II. Un algoritmo ganador tiene el valor de indicador más alto. Las tablas 2 y 3 resumen los resultados obtenidos de esta métrica para instancias con objetivos 3 a 15.

$$\text{Diff}\left(I_j^1, I_j^2\right) = \begin{cases} 100\left(\dfrac{I_j^1 - I_j^2}{I_j^1}\right), & \text{if } I_j^1 > I_j^2 \\ 100\left(\dfrac{I_j^2 - I_j^1}{I_j^2}\right), & de\ lo\ contrario \end{cases} \tag{1.26}$$

Instancia	**Diferencia (I^1_1, I^2_1)**	**Diferencia (I^1_2, I^2_2)**
o3p100	88%	11%
o4p100	69%	14%
o8p100	97%	17%
o9p100	97%	8%
o13p100	89%	17%
o15p100	96%	27%

Tabla 3. *Porcentaje de diferencia de promedio de cardinalidad de carteras no dominadas*

Instancia	Diferencia (I^1_3, I^2_3)	Diferencia (I^1_4, I^2_4)
o3p100	7%	13%
o4p100	14%	13%
o8p100	14%	14%
o9p100	5%	10%
o13p100	19%	15%
o15p100	24%	22%

Tabla 4. *Porcentaje de diferencia en cardinalidad mínima y máxima de carteras*

Los resultados de las Tablas 3 y 4 muestran que I-MOEA / D mejora todas las medidas de los indicadores con respecto a I-NSGA-II en rangos porcentuales que varían en [69, 97], [8, 27], [7, 24], y [10,13] para los indicadores I_1, I_2, I_3 e I_4, respectivamente. Estos resultados indican que I-MOEA/D obtiene más soluciones no dominadas y carteras con mayor número de proyectos, lo cual es deseable.

Finalmente, la Tabla 5 compara la proporción de dominancia por algoritmo. Para ello, un conjunto EP* combina los conjuntos finales EP_1 y EP_2; este nuevo conjunto es el frente final no dominado. Luego, calculamos el número de soluciones de EP_1 y EP_2 que aparecen en EP*. Observemos que EP_1 y EP_2 corresponden a los frentes finales no dominados EP de I-MOEA/D e I-NSGA-II, respectivamente. La columna 3 contiene el número de soluciones no dominadas que todavía aparecen en EP*. La columna 4 informa el número de soluciones que se dominaron después de la integración.

Instancia	Algoritmo	Total, de carteras No dominadas	Carteras dominadas
o2p100	**I-NSGA-II**	0	63
	I-MOEA/D	55	0
o3p100	**I-NSGA-II**	441	85
	I-MOEA/D	4556	0
o4p100	**I-NSGA-II**	0	139
	I-MOEA/D	449	0
o8p100	**I-NSGA-II**	1	584
	I-MOEA/D	21327	0
o9p100	**I-NSGA-II**	546	33
	I-MOEA/D	27863	0
o13p100	**I-NSGA-II**	0	521
	I-MOEA/D	5123	0
o15p100	**I-NSGA-II**	1	676
	I-MOEA/D	21417	0

Tabla 5. *Carteras dominadas entre I-MOEA/D e I-NSGA-II*

Teniendo en cuenta la información de la Tabla 5, todas las soluciones de I-MOEA/D permanecen no dominadas. Además, resulta que dominan varias soluciones proporcionadas por I-NSGA-II y, en algunos casos, todas (instancias 2 y 13). Estos resultados corroboran el

excelente desempeño de I-MOEA/D para resolver UPPS sobre I-NSGA-II con muchos objetivos.

El diseño experimental concluyó con un análisis de las diferencias estadísticas de los resultados observados. En particular, una prueba de Wilcoxon (1945) validó la diferencia en el indicador I_1. La muestra considerada fue el número de carteras no dominadas de cada una de las 30 ejecuciones de una instancia. La prueba utilizó un nivel de significancia del 5%. La hipótesis nula fue "H0 = Las medianas de las diferencias entre las dos muestras de grupo son iguales". La Tabla 6 resume los resultados.

Instancia	***p-value***	Resultado
o2p100	0.57746866	Se acepta H0
o3p100	0.04311445	Se rechaza H0
o4p100	0.04311445	Se rechaza H0
o8p100	0.04311445	Se rechaza H0
o9p100	0.04311445	Se rechaza H0
o13p100	0.04311445	Se rechaza H0
o15p100	0.04311445	Se rechaza H0

Tabla 6. *Prueba de Wilcoxon*

Los resultados de la Tabla 6 muestran que hay diferencias significativas en 6 casos y, según la información de la Tabla 1, las diferencias favorecen a I-MOEA/D. Señalemos que I-NSGA-II funcionó mejor que I-MOEA/D solo en la instancia "*o2p100*". Sin embargo, no existe una diferencia estadística significativa en su desempeño. En conclusión, el desempeño general de I-MOEA/D mejora en gran medida el de I-NSGA-II en las instancias seleccionadas de UPPS.

CAPITULO 6

Conclusiones

6.1 Conclusión

Este trabajo de investigación propone una nueva estrategia evolutiva llamada I-MOEA/D. Las principales características de este algoritmo son el uso de intervalos para expresar incertidumbre y manejar muchos objetivos. Una comparación en el desempeño entre I-MOEA/D e I-NSGA-II evaluó la relevancia de nuestro enfoque. En condiciones experimentales iguales en un ambiente controlado, los resultados muestran que I-MOEA/D supera a I-NSGA-II (Balderas, 2018), lo que demuestra la importancia de I-MOEA/D.

El I-MOEA/D requiere al menos modificar los operadores genéticos, el operador reparación/mejora, los métodos de actualización de los valores objetivos ideales y la población, con el fin de integrar adecuadamente el uso de intervalos. La estrategia requería la definición de algunos operadores de intervalo para realizar operaciones aritméticas, relacionales y de dominancia. El operador de dominancia aparece con la definición de operadores relacionales para comparar.

Los resultados observados muestran que I-MOEA/D e I-NSGA-II resuelven UPPS. Sin embargo, con el aumento de objetivos, el rendimiento de I-MOEA/D mejora el de I-NSGA-II, como se esperaba, particularmente en las instancias analizadas con un número de objetivos que varía de dos a quince. Los resultados muestran que, con un número creciente de objetivos, I-MOEA/D devuelve soluciones con mejor calidad.

6.2 Trabajos futuros

El número y la diversidad de soluciones que ofrece I-MOEA / D son grandes. Esta es una buena condición en contraste con I-NSGA-II porque significa que I-MOEA/D se aproxima mejor al frente de Pareto. Sin embargo, es interesante preguntar si el proceso de búsqueda de I-MOEA/D puede incluir las preferencias de DM. Si esto último es posible, entonces, se podría entregar al DM un conjunto más reducido de soluciones, en función de sus prioridades. Por tanto, la adecuada incorporación de preferencias en el proceso de búsqueda del I-MOEA / D representa un área de investigación atractiva para futuros desarrollos.

Bibliografía

Baker, J. E. (1985). Adaptive selection methods for genetic algorithms, Proc. International Conf. on Genetic Algorithms, ED. J.J. Grefenstette, Carnegie-Mellon University, Pittsburgh, PA, pp. 101-111.

Baker, J. E. (1987). Balancing diversity and convergence in genetic search. PhD thesis, Computer Science Dept., Vanderbilt University, Nashville, TN, USA.

Balderas, F., Fernandez, E., Gomez, C. and Cruz-Reyes, L. (2016) Metaheuristic robust optimization of project portfolios using an interval-based model of imprecisions, International Journal of Combinatorial Optimization Problems and Informatics 7(3) 101–118, Available at https://www.ijcopi.org/index.php/ojs/article/view/32.

Balderas, F. (2018). *Modelando la imprecisión del problema de cartera de proyectos con filosofía gris.* Tijuana, BC.

Balderas, F., Fernández, E., Gómez, C., Cruz, L., Rangel, N., Morales, M.: A grey mathematics approach for evolutionary mul-ti-objective metaheuristic of project portfolio selection, Springer, Cham, 379-388, 2018.

Bartlett, M., Frisch, A., Hamadi, Y., Miguel, I., Tarim, A., Unsworth, C.: The Temporal Knapsack Problem and Its Solution. CPAIOR 34-48 (2005).

Brindle, A. (1981). *Genetic algorithms for function optimization*. Edmonton: University of Alberta. Department of Computing Science.

Cagnina, L. (2010). *Optimización Mono y Multiobjetivo a través de una Heurística de Inteligencia Colectiva.* . San Luis, Argentina.

Carlsson, C., Fuller, R., Heikkila, M. and Majlender, P., A fuzzy approach to R&D portfolio selection, International Journal of Approximate Reasoning 44(2) 93–105, 2007 https://doi.org/10.1016/j.ijar.2006.07.003.

Carazo, A. F., Gómez, T., Molina, J., Hernández-Díaz, A. G., Guerrero, F. M., &

Caballero, R. (2010). Solving a comprehensive model for multiobjective project portfolio selection. *Computers & Operations Research*, 630–639.

Coello, A. (2011). Introducción a la computación evolutiva. Clase No.6. CINVESTAV-IPN http: //delta.cs.cinvestav.mx/~ccoello

Coello, A. (2012). Introducción a la computación evolutiva. Clase No.7. CINVESTAV-IPN http: //delta.cs.cinvestav.mx/~ccoello

Damghani, K., Sadi-Nezhad, S. and Aryanezhad, M., A modular decision support system for optimum investment selection in presence of uncertainty: Combination of fuzzy mathematical programming and fuzzy rule based system, Expert Systems with Applications 38(1) 824–834, 2011, http://doi.org/10.1016/j.eswa.2010.07.040.

Debasish, M., Abhijit, Ch. & Aparajita, S.(2020). *Chapter 8- Optimal and Robust Control*. 243-286. Power System Small Signal Stability Analysis and Control (Second Edition). https://doi.org/10.1016/B978-0-12-817768-6.00008-1

Deb, K. & Srinivas, N. (1994). Multiobjective optimization using nondominated sorting in genetic algorithms. Evolutionary computation 2(3), 221-248.

Deb, K., Agrawal, S., Pratap, A., and Meyarivan, T. (2000). A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II. In Proceedings of the 6th International Conference on Parallel Problem Solving from Nature, volume 1917 of Lecture Notes in Computer Science, Berlin, Heidelberg. Springer.

Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. A. M. T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE transactions on evolutionary computation, 6(2), 182-197.

De Jong, K. A. (1975). An analysis of the behavior of a class of genetic adaptive systems. PhD thesis, University of Michigan Ann Arbor, MI, USA.

Díaz, R. Optimización de estructuras reticuladas planas de madera mediante algoritmos genéticos. Valdivia, Chile, 2014.

Duarte, A., Pantrigo, J., & Gallego, M. (2007). *Metaheurísticas.* Universidad Rey Juan Carlos.

Edwards, A. (1954) Statistical Methods for the Behavioral Sciences. Nueva York: Holt.

Estévez, P. (1997), *Optimización mediante Algoritmos Genéticos*, Anales del Instituto de Ingenieros deChile, nº 1, pp. 83-92.

Fliedner, T. and Liesio, J., Adjustable robustness for multi-attribute project portfolio selection, European Journal of Op-erational Research 252 931–946, 2016, http://doi.org/10.1016/j.ejor.2016.01.058.

Flores, E., Miranda, M., & Villasís, M. (2017). El protocolo de investigación VI: cómo elegir la prueba estadística adecuadaa. Estadística inferencial. . *Rev Alerg Mex.*, 364-370.

Fox, G., & Baker, N. (1985). Project Selection Decision Making Linked to a Dynamic Environment. *Management Science*, 1272-1285.

Glover, F. (1986). Future paths for integer programming and links to artificial intelligence. *Computers and Operations Research*, 533-549.

Goldeberg, D. (1989). *Genetic Algorithms in Search, Optimization and Machine Learning.* Alabama: Addison-Wesley Publishing Company, Inc.

Goldberg, D. E. (1990). A note on Boltzmann tournament selection for genetic algorithms and population-oriented simulated annealing. Complex Systems, 4, 445-460.

Hasuike, T., Katagiri, H. and Ishii, H., Portfolio selection problems with random fuzzy variable returns, Fuzzy Sets and Systems 160(18), 2579–2596, 2009, https://doi.org/10.1016/j.fss.2008.11.010.

Holland J. (1975)Adaptation in natural and artificial systems: An introductory analysis with applications to biology, control, and artificial intelligence. U Michigan Press.

Huang, X., Optimal project selection with random fuzzy parameters, International Journal of Production Economics 106(2) 513–529, 2007, https://doi.org/10.1016/j.ijpe.2006.06.011.

Ishihuchi, H. and Tanaka, M. (1990), "Multiobjective programming in optimization of the Interval objective function", European Journal of Operation Research, Vol. 48, pp. 219-25

J.H. Holland (1975) Adaptation in Natural and Artificial Systems, University of Michigan Press,

J.P. Kelly and I.H. Osman. *Meta-Heuristic: Theory and Applications*. Kluwer Academic Publisher, 1996.

Ke, H. and Liu, B., Project scheduling problem with mixed uncertainty of randomness and fuzziness, European Journal of Operational Research 183(1) 135–147, 2007, http://doi.org/10.1016/j.ejor.2006.09.055.

Kuchta, D., Use of fuzzy numbers in project risk (criticality) assessment, International Journal of Project Management 19(5) 305–310, 2001, http://doi.org/10.1016/S0263-7863(00)00022-3.

Larousse. (2016). Diccionaro Manual de Sinonimos y Antónimos Vox. Larousse Editorial, S.L.

Liesio, J., Mild, P., and Salo, A. (2007), Preference programming for robust portfolio modeling and project selection, European Journal of Operational Research, 181(3), 1488–1505, http://doi.org/10.1016/j.ejor.2005.12.041.

Liesio, J., Mild, P. and Salo, A., (2008) Robust portfolio modeling with incomplete cost information and project interdependencies, European Journal of Operational Research 190(3), 679–695, https://doi.org/10.1016/j.ejor.2007.06.049.

Lin, D. y Wang, S. A multiobjective genetic algorithm for portfolio selection problem. 2001.

López, J. (2014). *Optimización Multi-objetivo Aplicaciones a problemas del mundo real.*

Medaglia,A., Graves, S. and Ringuest, J., A multiobjec-tive evolutionary approach for linearly constrained project se-lection under uncertainty, European Journal of Operational Research 179, 869–894, 2007, https://doi.org/10.1016/j.ejor.2005.03.068

Miller, Brad; Goldberg, David. (1995). Genetic Algorithms, Tournament Selection, and the Effects of Noise. Complex Systems 9: 193-212.

Mitchell, M. (1998). An introduction to genetic algorithms. MIT press.

Moore, R.E. (1979). Methods and applications of interval analysis. Society for Industrial and Applied Mathematics.

Moré, Iosvani (2017), Ajuste de funciones mediante metaheurísticas aplicado a la detección de fase en arreglos de metrología óptica, León, Guanajuato, México.

Morillo, D., Moreno, L., & Díaz, J. (2014). Metodologías Analíticas y Heurísticas para la Solución del Problema de Programación de Tareas con Recursos Restringidos (RCPSP): una revisión. Parte 2. *Universidad EAFIT*, 2013-227.

Olson, N. (2010). *Taken for Granted: The construction of order in the process of library Managmen System Decision Making.* University of Borås: Intellecta Infolog.

Osyczka, A. (1985) Multicriterion Optimization for Engineering Design, In: Gero, J.S. (ed):

Design Optimization. New York: Academic Press Inc., 193–227

Perez, F. and Gomez, T., Multiobjective project portfolio selec-tion with fuzzy constraints, Annals of Operations Research 245(1–2) 7–29, 2016 https://doi.org/10.1007/s10479-014-1556-z.

Q. Zhang and H. Li, "MOEA/D: A Multiobjective Evolutionary Algorithm Based on Decomposition," in IEEE Transactions on Evolutionary Computation, vol. 11, no. 6, pp. 712-731, Dec. 2007, doi: 10.1109/TEVC.2007.892759.

Rabbani, M., Bajestani, M. and Khoshkhou, G., A multi-objective particle swarm optimization for project selection problem, Expert Systems with Applications 37(1) (2010) 315–321, https://doi.org/10.1016/j.eswa.2009.05.056.

Rivera, G., Cruz, L., Fernandez, E., Gomez, C., & Perez, F. (2014). Many-objective portfolio optimization of interdependent projects with 'a priori' incorporation of decision-maker preferences. Applied Mathematics & Information, 8, 1517–1531.

Rivera, G. (2015). *Enfoque Metaherístico Híbrido para el Manejo de Muchos Objetivos en Optimización de Cartera de Proyectos Interdependientes con Decisiones de Apoyo Parcial.* Tijuana, Baja California.

Salas, G. (2019). *Ajuste de parámetros con lógica difusa aplicado a problemas de decisión.* Madero, Tamaulipas.

Salo, A., Keisler, J., & Morton, A. (2011). Portfolio Decision Analysis. Improved methods for resource allocation, International Series in Operations Research & Management Science. *Springer New York*, 162 3-27.

Schaffer, J.D. & Morishima, A. (1987). An Adaptive Crossover Distribution Mechanism for Genetic Algorithms and Their Applications: Proceedings of the Second International Conference on Genetic Algorithms, 36-40. Hillsdale, NJ: Lawrence Erlbaum Associates.

Sientific European Federation of Osteopaths (2014). *Las pruebas estadísticas* Madrid, España. Recuperado*:* *https://www.scientific-european-federation-osteopaths.org/las-pruebas-estadisticas/*

Sientific European Federation of Osteopaths (2019). Pruebas no paramétricas. Recuperado: https://www.scientific-european-federation-osteopaths.org/wp-content/uploads/2019/01/ALGUNAS-PRUEBAS-NO-PARAM%C3%89TRICAS.pdf

Sosa, H., Villagra,S., Villagra, A. (2013). Operadores de Mutación en Algoritmos Genéticos Celulares. ICT-UNPA, 86 141-157.

Syswerda. G. (1989). Uniform Crossover in Genetic Algorithms. Proceeding of the Third International Conference on Genetic Algorithms, 2-9. San Mateo, CA: Morgan Kaufmann.

Toppila, A. and Salo, A., Binary decision diagrams for gener-ating and storing non-dominated project portfolios with in-terval-valued project scores, European Journal of Operational Research 260(1) 244–254, 2017, http://doi.org/10.1016/j.ejor.2016.12.019.

Wilcoxon, F. (1945) Comparaciones individuales por métodos de clasificación. Boletín biométrico,80-83.

Wilfried, R., Xing, L., Dezhen, C. (2020). Chapter 6 - Optimal design of heat exchanger networks, Academic Press, 231-317, https://doi.org/10.1016/B978-0-12-817894-2.00006-6

Whitley, D. (1989). *The Genitor algorithm and selection pressure: Why rank-based allocation of reproductive trials is best*. Proceedings of the Third International Conference on Genetic Algorithms, 116-121.

Woeginger, G. (2003). Exact algorithms for NP-hard problems: A survey. *In Combinatorial Optimization—Eureka, You Shrink!*, 185-207.

Wu, M., Kwong, S., Yuheng, J., Ke, L., & Qingfu, Z. (2017). AdaptiveWeights Generation for Decomposition-Based Multi-Objective Optimization Using Gaussian Process Regression. GECCO, 641-648.

Yamaguchi, D., Li, G. D., Mizutani, K., Akabane, T., Nagai, M., & Kitaoka, M. (2006). On the generalization of grey relational analysis. Journal of Grey System, 9, 23–34

Young, R.C. (1931), "The algebra of many-valued quantities", Annals of Mathematics, Vol. 31, pp. 260-90.

Zhang, Q., & Hui, L. (2007). MOEA/D: A Multiobjective Evolutionary Algorithm Based on Decomposition. IEEE Transactions on Evolutionary Computation, 712-731.

ANEXO

A. Tabla de Áreas y ordenadas de la curva normal

(1) *Z* Puntuación típica $\left(\frac{x}{\sigma}\right)$	(2) *A* Área de la media $\alpha\left(\frac{x}{\sigma}\right)$	(3) *B* Área de la porción mayor	(4) *C* Área de la porción menor	(5) *Y* Ordenada $\alpha\left(\frac{x}{\sigma}\right)$
1.74	0.4591	0.9591	0.0409	0.0878
1.75	0.4599	0.9699	0.0401	0.0863
1.76	0.4608	0.9608	0.0392	0.0848
1.77	0.4616	0.9616	0.0384	0.0833
1.78	0.4625	0.9625	0.0375	0.0818
1.79	0.4633	0.9633	0.0367	0.0804
1.80	0.4641	0.9641	0.0359	0.0790
1.81	0.4649	0.9649	0.0351	0.0775
1.82	0.4656	0.9656	0.0344	0.0761
1.83	0.4664	0.9664	0.0336	0.0748
1.84	0.4671	0.9671	0.0329	0.0734
1.85	0.4678	0.9678	0.0322	0.0721
1.86	0.4686	0.9686	0.0314	0.0707
1.87	0.4693	0.9693	0.0307	0.0694
1.88	0.4699	0.9699	0.0301	0.0681
1.89	0.4706	0.9706	0.0294	0.0669
1.90	0.4713	0.9713	0.0287	0.0656
1.91	0.4719	0.9719	0.0210	0.0644
1.92	0.4726	0.9726	0.0274	0.0632
1.93	0.4732	0.9732	0.0268	0.0620
1.94	0.4738	0.9738	0.0262	0.0608
1.95	0.4744	0.9744	0.0256	0.0596
1.96	0.4750	0.9750	0.0250	0.0584
1.97	0.4756	0.9756	0.0244	0.0573
1.98	0.4761	0.9761	0.0239	0.0562
1.99	0.4767	0.9767	0.0233	0.0551

Tabla 7. Áreas y ordenadas de la curva normal en función de x/σ (Edwards, 1954)