TECNOLÓGICO NACIONAL DE MÉXICO
Instituto Tecnológico de Ciudad Madero

INSTITUTO TECNOLÓGICO DE CIUDAD MADERO
División de Estudios de Posgrado e Investigación

Estrategias de diversidad de las metaheurísticas A^2-NSGA-III y Búsqueda Dispersa Multiobjetivo para la solución del problema de Selección de Cartera de Proyectos.

OPCIÓN I:

Tesis profesional

Que para obtener el título de:

Maestro en Ciencias de la Computación

Presenta:

I. S. C. Daniel Adalberto Martínez Vega

Asesor:

D.C.C. Guadalupe Castilla Valdez

Cd. Madero, Tamaulipas Noviembre de 2015

SEP
SECRETARÍA DE EDUCACIÓN PÚBLICA

TECNOLÓGICO NACIONAL DE MÉXICO
Instituto Tecnológico de Ciudad Madero

"2015, Año del Generalísimo José María Morelos y Pavón"

Cd. Madero, Tamps; a 30 de Octubre de 2015.

OFICIO No.: U5.263/15
AREA: DIVISIÓN DE ESTUDIOS
DE POSGRADO E INVESTIGACIÓN
ASUNTO: AUTORIZACIÓN DE IMPRESIÓN DE TESIS

ING. DANIEL ADALBERTO MARTÍNEZ VEGA
NO. DE CONTROL G03070057
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, el cual está integrado por los siguientes catedráticos:

PRESIDENTE :	DRA. LAURA CRUZ REYES
SECRETARIO :	DR. JUAN JAVIER GONZÁLEZ BARBOSA
VOCAL :	DRA. GUADALUPE CASTILLA VALDEZ
SUPLENTE	DRA. CLAUDIA GUADALUPE GÓMEZ SANTILLÁN
DIRECTORA DE TESIS :	DRA. GUADALUPE CASTILLA VALDEZ
CO-DIRECTORA DE TESIS:	DRA. CLAUDIA GUADALUPE GÓMEZ SANTILLÁN

Se acordó autorizar la impresión de su tesis titulada:

"ESTRATEGIAS DE DIVERSIDAD DE LAS METAHEURÍSTICAS A²-NSGA-III Y BÚSQUEDA DISPERSA MULTIOBJETIVO PARA LA SOLUCIÓN DEL PROBLEMA DE SELECCIÓN DE CARTERA DE PROYECTOS"

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
"POR MI PATRIA Y POR MI BIEN"®

M. P. MARÍA YOLANDA CHÁVEZ CINCO
JEFA DE LA DIVISIÓN

S.E.P.
DIVISION DE ESTUDIOS
DE POSGRADO E
INVESTIGACION
I T C M

c.c.p.- Archivo
Minuta

MYCHC 'NLEO ' jar

Ave. 1° de Mayo y Sor Juana I. de la Cruz Col. Los Mangos, C.P. 89440 Cd. Madero, Tam.
Tel. (833) 357 48 20. e-mail: itcm@itcm.edu.mx
www.itcm.edu.mx

Tabla de contenido

Capítulo 1. Introducción……. 1

1.1 Definición del problema de Selección de Cartera de Proyectos…… 2

1.2 Formulación del problema de Selección de Cartera de Proyectos…… 3

1.3 Objetivos……..... 6

1.3.1 Objetivo general…….……. 6

1.3.2 Objetivos específicos……..... 6

1.4 Justificación…..…… 7

1.5 Alcances……... 7

1.6 Limitaciones……. 7

Capítulo 2. Marco teórico…….……..…… 8

2.1 Optimización multiobjetivo…….…….…..... 8

2.1.1 Optimalidad de Pareto.……..…..……...……..... 8

2.1.2 Dominancia de Pareto…….…..…..……..... 9

2.2 Frente de Pareto……...…… 10

2.3 Modelos de preferencia……. 10

2.4 Método Electre III.……..... 11

2.4.1 Conjuntos difusos…….……..... 12

2.4.2 Relación de sobreclasificación difusa..……..... 12

2.4.3 Concordancia, Discordancia y Credibilidad……..…….…..... 13

2.4.3.1 Índice de Concordancia para cada criterio……..... 13

2.4.3.2 Índice de Concordancia global……..... 14

2.4.3.3 Índice de Discordancia…….……..... 14

2.4.3.4 Umbrales difusos……..……... 15

2.4.4 Cálculo de índices y grado de credibilidad en el modelo Electre III……. 16

2.5 Sistema relacional de preferencias……..……..... 18

2.5.1 Sistema relacional de preferencias difusas del Dr. Eduardo Fernández… 19

2.6 Definición de Heurística……. 21

2.7 Definición de Metaheurística……. 21

2.8 Complejidad del problema de Selección de Cartera de Proyectos…….. 21

2.9 Indicadores de desempeño……. 26

Capítulo 3. Estado del arte…….……..…… 29

Capítulo 4. Metaheurísticas multiobjetivo.……..…… 34

4.1 Introducción a los Algoritmos Genéticos……..... 35

4.2 Non-dominated Sorting Genetic Algorithm II (NSGA-II)…… 38

4.2.1 Descripción del método NSGA-II……..... 39

4.2.2 Estrategias de diversidad de la metaheurística NSGA-II..... 40

4.2.3 Algoritmo NSGA-II……... 42

4.2.4 Evaluación experimental del algoritmo NSGA-II……. 43

4.2.5 Resultados y conclusiones……. 44

4.3 NSGA-III, Adaptive NSGA-III (A-NSGA-III) y Efficiently Adaptive NSGA-III (A^2-NSGA-III)……. 46

4.3.1 Estrategias de diversidad de las metaheurísticas NSGA-III, A-NSGA-III y A^2-NSGA-III…….. 47

4.3.2 Descripción de los métodos NSGA-III, A-NSGA-III y A^2-NSGA-III…….. 49

4.3.3 Algoritmos NSGA-III, A-NSGA-III y A^2-NSGA-III……. 52

4.3.4 Evaluación experimental de los algoritmos NSGA-III y A^2-NSGA-III…... 53

4.3.5 Resultados y conclusiones……. 54
4.4 Introducción a la Búsqueda Dispersa Multiobjetivo…..……..... 57
4.4.1 Descripción del método Búsqueda Dispersa Multiobjetivo……... 61
4.4.2 Estrategias de diversidad de la Búsqueda Dispersa Multiobjetivo…… 61
4.5 Búsqueda Dispersa Multiobjetivo Guiada por Preferencias..... 65
4.5.1 Estrategias de diversidad de MSS-GP…..…….. 66
4.5.2 Algoritmo MSS-GP……..…….. 66
4.5.3 Evaluación experimental de los algoritmos MSS y MSS-GP….……... 70
4.5.3.1 MSS y A^2-NSGA-III utilizando las instancias estándar de la familia DTLZ 71
4.5.3.2 MSS-GP y A^2-NSGA-III utilizando las instancias del problema de Selección de Cartera de Proyectos……. 74
4.5.3.3 MSS y MSS-GP utilizando las instancias del problema de Selección Cartera de Proyectos……..... 75
4.5.3.4 MSS y NO-ACO utilizando las instancias del problema de Selección Cartera de Proyectos……..... 75
4.5.4 Resultados y conclusiones…….. 76

Capítulo 5. Conclusiones y futuras líneas de investigación…… 81
5.1 Conclusiones……...…….…….. 81
5.2 Trabajos futuros……..…...…….……... 83
Referencias Bibliográficas…..……..……... 84

Índice de Imágenes

2.1	Dominancia……	26
2.2	Dispersión (Spread)……	27
4.1	Ciclo de un Algoritmo Genético básico…….	36
4.2	Distancia de Crowding……...	41
4.3	Evolución NSGA-II=>A^2-NSGA-III…….	46
4.4	Solo 28 de 91 puntos de referencia encontraron una solución Pareto óptima para asociarse…….	47
4.5	Enfoque para añadir nuevos puntos de referencia en A-NSGA-III……...	48
4.6	Enfoque para añadir nuevos puntos de referencia en A^2-NSGA-III……..	48
4.7	DTLZ1……	71
4.8	DTLZ2……	72
4.9	DTLZ3……	73
4.10	DTLZ7……	74

Índice de Algoritmos

4.1	Distancia de Crowding……	41
4.2	NSGA-II……..	42
4.3	Algoritmo de las metaheurísticas NSGA-III, A-NSGA-III y A^2-NSGA-III..	52
4.4	Metaheurística SS……	60
4.5	Añadir solución al archivo externo…….	64
4.6	Método para la generación de P……..	67
4.7	Reinicialización de P……	69

Índice de Tablas

1.1 $f_{i,j}$…… 4

2.1 Categorías de preferencias……... 20

3.1 Trabajos relacionados…….. 33

4.1 Resultados de la experimentación para la instancia de tres objetivos mediante el indicador de desempeño de Dominancia…….. 44

4.2 Resultados de la experimentación para la instancia de nueve objetivos mediante el indicador de desempeño de Dominancia……. 45

4.3 Parámetros empleados en Electre III y MPDF para la instancia de tres objetivos…….. 53

4.4 Parámetros empleados en Electre III y MPDF para la instancia de nueve objetivos…….. 53

4.5 Resultados de la experimentación para la instancia de tres objetivos mediante el indicador d desempeño de Dominancia aplicando preferencias. 54

4.6 Resultados de la experimentación para la instancia de nueve objetivos mediante el indicador d desempeño de Dominancia aplicando preferencias. 55

4.7 Comparación de métricas sobre resultados en el problema DTLZ1 con A^2-NSGA-III y MSS……... 76

4.8 Comparación de métricas sobre resultados en el problema DTLZ2 con A^2-NSGA-III y MSS……... 77

4.9 Comparación de métricas sobre resultados en el problema DTLZ3 con A^2-NSGA-III y MSS……... 77

4.10 Comparación de métricas sobre resultados en el problema DTLZ7 con A^2-NSGA-III y MSS……... 77

4.11 Comparación mediante el indicador de desempeño de Dominancia sobre las soluciones devueltas por los algoritmos A2-NSGA-III y MSS-GP para las instancias del problema de Selección de Cartera de Proyectos……. 78

4.12 Comparación MSS vs MSS-GP…….. 79

5.1 Comparación de calidad entre A2-NSGA-III y MSS……. 82

Capítulo 1

Introducción

Debido a las necesidades de resolver problemas en donde los métodos numéricos no han tenido éxito, han surgido los llamados algoritmos evolutivos. Éstos se han utilizado ampliamente para resolver problemas complicados de optimización. Los algoritmos evolutivos surgieron de la abstracción biológica de la evolución de los seres vivos tomando como base la teoría de "el origen de las especies" de Charles Darwin y basando su estructura en la propuesta del Neodarwinismo.

En este trabajo se propone el desarrollo de las metaheurísticas A^2-NSGA-III y Búsqueda Dispersa Multiobjetivo (MSS, por su sigla en inglés) para la solución del problema de Selección de Cartera de Proyectos con el fin de estudiar las estrategias de diversidad incorporadas en cada uno.

La metaheurística A^2-NSGA-III se utiliza para resolver problemas de optimización de muchos-objetivos, surge debido a que la mayoría de los problemas reales requieren soluciones a problemas con más de 10 objetivos, por otro lado, los algoritmos de optimización multiobjetivo evolutivos ofrecen buenos resultados solo para problemas con menos de 4 objetivos [Deb K. y Jain H., 2014]. Esta metodología deriva de una serie de estrategias incorporadas al método NSGA-II. Este método primero tuvo una extensión en su estructura convirtiéndose en MO-NSGA-II o NSGA-III, en donde modifica la forma de proporcionar diversidad a las soluciones utilizando un conjunto de puntos de referencia que guían a las soluciones en vez del clásico método de "Distancia de Crowding" originalmente aplicado en NSGA-II. Posteriormente se volvió una metodología adaptativa llamada A-NSGA-III que se distingue por la eliminación e inclusión de nuevos puntos de referencia sobre la marcha.

Finalmente A^2-NSGA-III hace más eficiente el método de adición de puntos de referencia [Jain H. y Deb K., 2013].

La Búsqueda Dispersa (SS, por su sigla en inglés) se basa en combinar soluciones de calidad con soluciones diversas, a partir de un conjunto denominado conjunto de referencia. Dicha técnica fue propuesta en los años 70's por Fred Glover [Glover F., 1977] y utiliza estrategias para diversificar e intensificar la búsqueda y así obtener nuevas buenas soluciones. Esta metaheurística ha sido utilizada para problemas multiobjetivo debido a que trabaja sobre un conjunto de soluciones. SS se ha adaptado con éxito para la solución de problemas estándar multiobjetivo tales como la familia de problemas ZDT [Nebro A. et. al, 2007].

1.1 Definición del problema de Selección de Cartera de Proyectos

Una de las principales tareas de dirección en las organizaciones del sector público, fundaciones, centros de investigación y empresas que realizan investigación y desarrollo, consiste en evaluar un conjunto de proyectos que compiten por apoyo financiero, y seleccionar aquellos que aporten el máximo beneficio a la organización, constituyendo con este subconjunto una cartera de proyectos [Nebro A. et. al, 2007].

La modelación de los problemas de Selección de Cartera de Proyectos se basa en las siguientes premisas:

a) Existe un conjunto bien definido de N proyectos, cada uno de ellos perfectamente caracterizado desde el punto de vista de los beneficios económicos que puede aportar y de sus requerimientos presupuestales.
b) Se trata de decidir qué subconjunto de proyectos conforman la cartera ideal, de modo que se optimice una cierta medida de calidad. Si se prescinde de incertidumbre y riesgo, y se conoce el beneficio que genera cada proyecto, se intenta maximizar el valor actual neto de cada beneficio asociado con la cartera [Davis R. y McKeown P., 1984]. Se asume que si un proyecto es aceptado en la cartera recibirá todo el apoyo que solicita.

1.2 Formulación del problema de Selección de Cartera de Proyectos

En todo problema de decisión el encargado de tomar la decisión final es conocido como Tomador de Decisiones. Este elemento central es una persona (o grupo), cuyo sistema de preferencias es determinante en la solución de problemas que consideran varios objetivos, los cuales posiblemente se encuentren en conflicto entre sí [Sánchez P., 2012].

Sean N proyectos de interés social que cumplen determinados requisitos mínimos de aceptabilidad para ser apoyados. Cada proyecto tiene asociados a diferentes regiones, un área y un costo:

- $A = \langle a_1, a_2, \dots, a_k \rangle$ Áreas
- $G = \langle g_1, g_2, \dots, g_r \rangle$ Regiones
- $C = \langle c_{1,} c_{2,} , \dots, c_N \rangle$ Costos

En donde c_j es una cantidad de dinero que satisface plenamente los requisitos presupuestales del proyecto j [Jain H. y Deb K., 2013].

Sea $X = \langle x_1, x_2, \dots, x_N \rangle$, el conjunto de N proyectos donde:

$$x_i = \begin{cases} 1 & \text{si el } i-\text{ésimo proyecto es soportado} \\ 0 & \text{en otro caso} \end{cases} \tag{1.1}$$

Una de las tareas más complejas es la evaluación de los proyectos, para la cual se considera la aportación que cada proyecto tiene sobre cada uno de los objetivos planteados por la institución que otorga los recursos económicos.

De tal manera que cada proyecto x_i se puede representar de acuerdo al nivel de aportación a cada objetivo (beneficio), mediante un vector $f_{x_i} = \langle f_{1x_i}, f_{2x_i}, \dots, f_{px_i} \rangle$ llamado vector beneficio para el proyecto i considerando p objetivos.

Una matriz que contiene el conjunto de todos los vectores beneficio del total de los n proyectos, constituirán la matriz de beneficios f, de dimensión $p \times N$.

Tabla 1.1. $f_{i,j}$

Proyecto / Objetivo	**1**	**2**	**…**	***N***
1	$f_{1,1}$	$f_{2,1}$	…	$f_{N,1}$
2	$f_{1,2}$	$f_{2,2}$	…	$f_{N,2}$
.	.	.	.	.
p	$f_{1,p}$	$f_{2,p}$	…	$f_{N,p}$

En donde p es el número de objetivos. Cada componente del vector f_{x_i} indica la contribución del proyecto i al $h-ésimo$ objetivo ($f_{h,i}$).

Sea P el monto total de los recursos financieros disponibles para distribuir a los diferentes proyectos. Dado que cada proyecto tiene un costo c_i, cualquier cartera de proyectos debe cumplir con la siguiente restricción presupuestal:

$$\left(\sum_{i=1}^{N} x_i \, c_i\right) \leq P \tag{1.2}$$

Asumamos como posible que existan restricciones presupuestales por cada área de inversión. De manera que si P_l es el presupuesto dedicado al área l, y existe un presupuesto mínimo $P_{l_{min}}$ y un presupuesto máximo $P_{l_{max}}$ establecidos tales que:

$$P_{l_{min}} \leq \; P_l \leq P_{l_{max}} \tag{1.3}$$

La restricción de presupuesto por área que debe cumplir cada cartera está dada por:

$$P_l = \sum_{i=1}^{N} x_i \, c_i \, a_{i_l} \tag{1.4}$$

En donde a es una variable binaria que indica si el proyecto i pertenece al área socioeconómica l.

Por otro lado cada proyecto beneficia a una región en particular y al igual que con las áreas, existe un límite presupuestario inferior $P_{r_{min}}$ y un presupuesto máximo $P_{r_{max}}$ establecidos tales que:

$$P_{r_{min}} \leq P_r \leq P_{r_{max}} \tag{1.5}$$

La restricción de presupuesto por región que debe cumplir cada cartera está dada por:

$$P_r = \sum_{i=1}^{N} x_i \, c_i \, g_{i_r} \tag{1.6}$$

En donde *g* es una variable binaria que indica si el proyecto *i* pertenece a la región *r* o no.

La calidad de una cartera X depende de los beneficios que aportan los proyectos que la integran y se representa mediante el vector de calidad $z(X)$, cuyas componentes son al mismo tiempo valores de calidad en relación a cada uno de los *p* objetivos de los proyectos:

$$z(X) = \sum_{j=1}^{p} z_j(X) \tag{1.7}$$

En donde:

$$z_j(X) = \sum_{i=1}^{N} x_i f_{j,i} \tag{1.8}$$

Siendo f la matriz de beneficios cuyas filas representan cada uno de los p objetivos y sus columnas cada uno de los N proyectos.

Sea S_F el espacio de carteras factibles, la solución del problema de Selección de Cartera de Proyectos consiste en encontrar una o más carteras que satisfagan la ecuación 1.9.

$$\max_{x \in S_F} \{z(X)\} \tag{1.9}$$

Es decir, que las únicas soluciones aceptadas serán las que cumplan con las restricciones establecidas por las ecuaciones (1.2), (1.3) y (1.5).

1.3 Objetivos

Este proyecto de investigación forma parte de un conjunto de proyectos que se abordan en una red de investigadores para el estudio del problema de Selección de Cartera de Proyectos denominada Red de Optimización y Apoyo a la Decisión, la cual está formada por la Universidad Autónoma de Sinaloa (UAS), Universidad Autónoma de Nuevo León (UANL) y el Instituto Tecnológico de Ciudad Madero (ITCM).

La colaboración por parte del ITCM está enfocada en el desarrollo de algoritmos de optimización multiobjetivo (para nueve o más) que permitan resolver el problema Selección de Cartera de Proyectos [Bastiani S., 2013].

1.3.1 Objetivo general

Desarrollar dos métodos que den solución al problema de Selección de Cartera de Proyectos basados en las metaheurísticas A^2-NSGA-III y MSS, con el fin de realizar un estudio de las estrategias de diversidad mediante distintos indicadores de desempeño.

1.3.2 Objetivos específicos

Realizar el diseño de dos metaheurísticas, MSS y A^2-NSGA-III, enfocados en dar solución al problema de Selección de Cartera de Proyectos.

Llevar a cabo una selección de los mejores parámetros de las metaheurísticas.

Implementación y evaluación experimental de ambas metaheurísticas que ofrezca una solución al problema de Selección de Cartera de Proyectos.

Llevar a cabo un estudio de las estrategias de diversidad de ambas metaheurísticas apoyado en indicadores de desempeño.

1.4 Justificación

El problema de Selección de Cartera de Proyectos pertenece a la categoría de NP-Completos, para los cuales no es factible implementar una solución exacta, cuya solución alternativa se puede obtener con la aplicación de algoritmos metaheurísticos.

Los algoritmos A^2-NSGA-III y MSS han mostrado su eficiencia en la solución de problemas multiobjetivo estándar, sin embargo no se tiene evidencia sobre la aplicación de estos algoritmos para la solución del problema de Selección de Cartera de Proyectos, situación que motiva a profundizar en su estudio con la incorporación de estrategias de diversificación enfocadas a la mejora del desempeño [Deb K. y Jain H., 2014; Nebro A. et. al, 2008].

1.5 Alcances

En este trabajo se aborda el problema de Selección de Cartera de Proyectos que son clasificables en áreas y regiones, este modelo incorpora límites mínimos y máximos presupuestales en cada tipo de área.

1.6 Limitaciones

Esta propuesta no incluye entre sus objetivos el manejo de apoyo parcial ni calendarización.

Las instancias de prueba que se utilizarán para evaluar las metaheurísticas propuestas son las reportadas por Cruz Reyes en [Cruz-Reyes L. et. al, 2014].

CAPÍTULO 2

Marco teórico

2.1 Optimización multiobjetivo

La optimización multiobjetivo se define 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. Esas funciones forman una descripción matemática de los criterios a optimizarse y generalmente se encuentran en conflicto entre sí. Por lo tanto, el término optimizar significa encontrar las soluciones que producen valores aceptables para todas las funciones objetivo" [Osyczka A., 1985].

2.1.1 Optimalidad de Pareto

En problemas multiobjetivo, cuando los objetivos se encuentran en conflicto entre sí, se buscan normalmente soluciones compromiso en vez de una única solución. Por lo tanto, la noción de óptimo es diferente en estos casos. La noción de óptimo más comúnmente adoptada es la propuesta por Francis Ysidro Edgeworth, la cual más tarde fue generalizada por Vilfredo Pareto, dicha noción es comúnmente conocida bajo el término de optimalidad de Pareto [Sánchez P., 2012].

Se dice que un vector de variables de decisión: $\overrightarrow{x^*} \in F$ es un óptimo de Pareto existente en la región factible F si se cumplen las siguientes condiciones matemáticas (asumiendo minimización):

$$f_i\left(\overrightarrow{x^*}\right) \leq f_i(\vec{x}), \qquad \forall i \in [1, 2, \ldots, k] \qquad \text{y} \qquad (2.1)$$

$$f_i\left(\overrightarrow{x^*}\right) < f_i(\vec{x}), \qquad \exists i \in [1, 2, \ldots., k]$$

En palabras, $\overrightarrow{x^*}$ es un óptimo de Pareto si no existe otro vector factible de variables de decisión $\vec{x} \in F$ que mejore algún criterio sin causar el deterioro simultáneo de al menos otro criterio. Sin embargo, este concepto casi siempre ofrece no una, sino varias soluciones llamadas Conjunto de óptimos de Pareto (P^*). Los vectores $\overrightarrow{x^*}$ correspondientes a las soluciones incluidas en el conjunto de óptimos de Pareto son llamados no-dominados. La imagen del conjunto de óptimos de Pareto bajo las funciones objetivo es llamada Frente de Pareto.

2.1.2 Dominancia de Pareto

En algoritmos multiobjetivo se utiliza frecuentemente el concepto de dominancia de Pareto al comparar dos soluciones y decidir si una domina a otra o no. Una solución $\overrightarrow{x_a}$ se dice que domina a otra $\overrightarrow{x_b}$ si se cumplen las siguientes condiciones (para el caso de minimización) [Sánchez P., 2012]:

La solución $\overrightarrow{x_a}$ no es peor que $\overrightarrow{x_b}$ en todos los objetivos:

$$f_i(\overrightarrow{x_a}) \leq f_i(\overrightarrow{x_b}), \qquad \forall i \in [1, 2, \dots, k] \quad \text{y} \tag{2.2}$$

1. La solución $\overrightarrow{x_a}$ es estrictamente mejor que $\overrightarrow{x_b}$ en al menos un objetivo:

$$f_i(\overrightarrow{x_a}) < f_i(\overrightarrow{x_b}), \qquad \exists i \in [1, 2, \dots, k] \tag{2.3}$$

Si alguna de las condiciones (2.2) o (2.3) son violadas, la solución $\overrightarrow{x_a}$ no domina a la solución $\overrightarrow{x_b}$. Es decir que para que una solución domine a otra, ésta necesita ser estrictamente mejor en al menos un objetivo, y no peor en ninguno de ellos. Esto es, al comparar dos soluciones A y B, sólo pueden existir tres posibles soluciones:

- A domina a B.
- A es dominada por B.
- A y B no se dominan (son no dominadas entre sí).

2.2 Frente de Pareto

A la representación de las funciones objetivo cuyos vectores son no dominados y además están en el conjunto de óptimos de Pareto se le llama Frente de Pareto [Sánchez P., 2012].

Definición formal:

Para un problema multiobjetivo dado $F(\vec{x})$ y un conjunto de óptimos de Pareto P^*, el frente de Pareto FP^* se define como:

$$FP^* = \{F(\vec{\mathrm{x}}) = (\mathrm{f}_1(\vec{\mathrm{x}}), \mathrm{f}_2(\vec{\mathrm{x}}), \dots, \mathrm{f}_\mathrm{k}(\vec{\mathrm{x}}))\,|\,\vec{\mathrm{x}} \in P^*\} \tag{2.4}$$

De forma general, no existe un método eficiente para encontrar el frente de Pareto, y la mejor forma de hacerlo es probar todos y cada uno de los puntos en la zona factible. Obviamente, en muchas ocasiones el espacio de búsqueda es tan grande que un proceso enumerativo es incosteable y de ahí se origina la necesidad de utilizar heurísticas para producir aproximaciones del frente de Pareto de un problema de optimización.

2.3 Modelos de preferencia

Un modelo de preferencia es un paso inevitable en una variedad de campos: la economía, la sociología, la programación matemática, incluso la medicina, la arqueología, ciencias de la computación y el análisis de decisiones.

Los científicos construyen modelos para comprender y representar mejor una situación dada; tales modelos también pueden usarse para propósitos operacionales. A menudo se da el caso en que es necesario comparar objetos en tales modelos, básicamente con el fin de establecer bien si hay un orden entre los objetos o para establecer si tales objetos están cerca. Los objetos pueden ser de todo tipo, por ejemplo intervalos de tiempo a partir de códigos computacionales para los sistemas de producción. Esta es la razón por la cual se usan los modelos de preferencia en una gran variedad de campos [Öztürk M. et. al, 2005].

En este trabajo de investigación se utilizan los modelos de preferencia con el propósito de ayudar a la toma de decisiones.

Si se considerara el caso de alguien (tomador de decisiones) que intenta comparar objetos tomando en cuenta diversos puntos de vista, se denota el conjunto de alternativas A^1 (a,b,c,…) y el conjunto de puntos de vista J (j =1,2,…, m). En este caso, un dato $g_j(a)$ corresponde a la evaluación de la alternativa a del punto de vista $j \in J$.

Como ya se mencionó, comparar dos objetos puede ser visto como en busca de una de las dos siguientes situaciones posibles:

- Objeto a esta "antes" que el objeto b, donde "antes" implica una clase de orden entre a y b, tal orden designa indistintamente una preferencia directa (a se prefiere sobre b) o la inducción de una medición y su escala asociada (a ocurre antes que b, a es más larga, más grande, más segura, que b).
- Objeto a esta "cerca" del objeto b, en donde "cerca" puede ser considerado o bien como indiferencia (el objeto a y el objeto b arrojan resultados semejantes en el mismo propósito), o como una similaridad, o de nuevo puede ser inducido como una medición (a ocurre simultáneamente con b, ellos tienen la misma longitud, peso, seguridad, etc.).

Desde el punto de vista tradicional de la ayuda a la toma de decisiones nos centraremos en la primera situación. Ordenar relaciones es la base natural para la solución de la clasificación o de la elección de problemas. La segunda situación se asocia tradicionalmente a problemas en los que el objetivo es ser capaz de reunir los objetos que comparten una característica común con el fin de formar clases "homogéneas" o categorías (un problema de clasificación).

2.4 Método Electre III

A diferencia del Electre I, en Electre III la sobreclasificación se basa en conjuntos difusos.

Electre III determina el índice de concordancia entre las alternativas por cada criterio utilizando nuevos conceptos: el umbral difuso de indiferencia y el umbral difuso de preferencia, para redefinir las relaciones de preferencia.

Este método implementa el umbral difuso de veto para determinar el índice de discordancia el cual permite rechazar la premisa que una alternativa supera a la otra. De acuerdo a la

literatura los umbrales descritos anteriormente son difíciles de valorar, pero se pueden obtener arbitrariamente de acuerdo al conocimiento experto o por medio de encuestas.

Para determinar la preferencia entre alternativas se combinan las matrices de concordancia y discordancia para crear el índice de credibilidad, el cual valora para intensidad de la premisa de que alternativa es al menos tan buena como la otra. Este proceso permite establecer la clasificación de las alternativas [Smith Q. et. al, 2000].

2.4.1 Conjuntos difusos

La incorporación de vaguedad e imprecisión como fuente de incertidumbre en la solución de problemas multicriterio, ha sido un aspecto que tradicionalmente fue tratado mediante los análisis de sensibilidad. Sin embargo, la aparición de un cuerpo axiomático de la lógica borrosa o difusa en 1965, permitió incorporar, aunque de modo lento, técnicas más adecuadas para el tratamiento de estas cuestiones. Es así como una de las primeras formas abordadas de incertidumbre mediante preferencias borrosas, es la correspondiente a Electre III [Smith Q. et. al, 2000].

2.4.2 Relación de sobreclasificación difusa

Roy describe situaciones donde se presenta este comportamiento no ideal en el DM [Roy B., 1996].

Debido a esto usa un sistema preferencial compuesto de varias relaciones binarias. A continuación se describen dichas relaciones:

1. **Indiferencia**. Corresponde a la existencia de razones claras que justifican una equivalencia entre las dos alternativas. Se denota como xIy.
2. **Preferencia estricta**. Corresponde a la existencia de razones claras que justifican una preferencia significativa en favor de alguna de las dos alternativas. La declaración "x es estrictamente preferida sobre y" se denota como xPy.
3. **Preferencia débil**. Corresponde a la existencia de razones claras para preferir a x sobre y, pero que no son lo suficientemente significativas como para justificar una preferencia

estricta. Se utiliza cuando la indiferencia y la preferencia estricta no pueden ser claramente distinguidas. Es denotada como xQy.

4. **Incomparabilidad**. Ninguna de las situaciones anteriores predomina. Esto es porque faltan razones claras que justifiquen cualquiera de las relaciones anteriores. Se denota como xRy.

5. **Sobreclasificación**. Corresponde a la existencia de razones que justifican la declaración "x es al menos tan bueno como y", pero sin una división claramente establecida entre preferencia estricta, preferencia débil o indiferencia. La sobreclasificación se denota como xSy.

6. **k-Preferencia**. Corresponde a la existencia de razones claras que justifican ya sea una preferencia estricta o una incomparabilidad, pero sin existir una diferencia establecida entre ambas situaciones. Se denota como xKy.

7. **No preferencia**. Corresponde a la situación en la cual tanto la indiferencia como la incomparabilidad son posibles, sin existir una diferencia clara entre ellas. Se denota como $x \sim y$.

2.4.3 Concordancia, Discordancia y Credibilidad

Electre III define $\sigma(x, y)$, la credibilidad de que x sea al menos tan buena como y, como:

$$\sigma(x, y) = c(x, y) * d(x, y) \tag{2.5}$$

en donde $c(x, y)$ es el índice de concordancia y $d(x, y)$ el índice de discordancia [Gento A. y Redondo A., 2005].

2.4.3.1 Índice de Concordancia para cada criterio

Gento y Redondo definen una alternativa i como mejor que una alternativa k para el criterio j a partir del índice de discordancia $c_j\ (a_i, a_k)$ [Gento A. y Redondo A., 2005], que se define como:

$$\begin{aligned} c_j(a_i, a_k) = 0 &\Leftrightarrow p_j < g_j(a_k) - g_j(a_i) \\ 0 < c_j) < 1 &\Leftrightarrow q_j < g_j(a_k) - g_j(a_i) \leq p_j \\ c_j(a_i, a_k) = 1 &\Leftrightarrow g_j(a_k) - g_j(a_i) \leq q_j \end{aligned} \tag{2.6}$$

2.4.3.2 Índice de Concordancia global

Se define el índice de concordancia global C_{ik}, que indica cuando la alternativa i sobreclasifica a la alternativa k, a partir de la fórmula:

$$C_{ik} = \frac{\sum_{j=1}^{m} p_j \ . c_j(a_i, a_k)}{\sum_{j=1}^{m} p_j} \tag{2.7}$$

Los índices globales forman una matriz $(i \times k)$ [Gento A. y Redondo A., 2005].

2.4.3.3 Índice de Discordancia

En Electre III se utiliza el umbral de veto (v_j), que, por definición, es el valor de la diferencia entre $g_j(a_k) - g_j(a_i)$ a partir de la cual es prudente rechazar la sobreclasificación [Molica F. et. al, 2011].

Cuando se supera el valor del umbral de veto se rechaza la sobreclasificación, independientemente de lo que ocurra en el resto de criterios. La construcción del índice de discordancia es la siguiente:

$$\begin{gathered} d_j(a_i, a_k) = 0 \Leftrightarrow g_j(a_k) - g_j(a_i) \leq p_j \\ 0 < d_j(a_i, a_k) < 1 \Leftrightarrow p_j < g_j(a_k) - g_j(a_i) \leq v_j \\ d_j(a_i, a_k) = 1 \Leftrightarrow v < g_j(a_k) - g_j(a_i) \end{gathered} \tag{2.8}$$

Para cada par de alternativas (a, b) existe una medida de concordancia y una de discordancia. El paso final en la fase de construcción del modelo es combinar estas dos medidas para producir una medida del grado de sobreclasificación; es decir, un índice de credibilidad que avala la fuerza de la afirmación de que "a es por lo menos tan bueno como b". El grado de credibilidad para cada par de alternativas (a, b) es definido como:

$$S(a,b) = \begin{cases} c(a,b), \text{si } d_j(a,b) \leq c(a,b) \\ c(a,b) * \prod_{j \in J}(a,b) \ \frac{1 - d_j(a,b)}{1 - c(a,b)} \end{cases} \tag{2.9}$$

Donde, $J(a, b)$ es el conjunto de criterios que satisfacen $d_j(a, b) > c_j(a, b)$.

Esta fórmula presume que si la fuerza de concordancia sobrepasa la de discordancia, el valor de concordancia no debe ser modificado. En caso contrario, es forzado a cuestionar la afirmación de que el aSb y modificar $c(a,b)$ de acuerdo con la ecuación anterior. Si la discordancia es de 1,0 para cualquier (a,b) y cualquier criterio j, entonces no se tiene confianza de que aSb, por tanto, $S(a,b) = 0{,}0$. Esto concluye la construcción del modelo de sobreclasificación.

La definición de discordancia requiere de la introducción de un umbral de veto $v_j\big(g_j(a)\big)$ (función de $g_j(a)$) para cada criterio g_j, tal que la credibilidad de que a sea de categoría superior a b, es rechazada si:

$$g_j(a) \geq g_j(b) + v_j(g_j(b)) \tag{2.10}$$

Inclusive si es que todos los criterios restantes están en favor de que a, sea de categoría superior a b.

2.4.3.4 Umbrales difusos

La modelización de las preferencias del decisor en la versión Electre III es menos rígida, ya que se toma como punto de partida el siguiente argumento: si la diferencia entre las valoraciones de las alternativas a_j y a_k es muy pequeña, ¿el decisor continuará prefiriendo una de ellas?; ¿es esa pequeña diferencia razón suficiente para hacer más preferida una que la otra? El método Electre III consigue esta flexibilización con la introducción de tres nuevos umbrales, definidos para cada uno de los criterios considerados [García L. y Muñoz A., 2009]:

a) El umbral de preferencia (p), que sería la magnitud en que exigimos que la valoración de la alternativa a_j sea mayor que la de la alternativa a_k para poder hablar de preferencia fuerte de la primera respecto a la segunda.

b) El umbral de indiferencia (q), que sería la magnitud en que permitimos que la valoración de la alternativa a_k sea menor que la de la alternativa a_j para seguir siendo indiferentes.

c) El umbral de veto (v), que sería la magnitud de la diferencia $(a_j - a_k)$ a partir de la cual nunca se aceptará que la alternativa k supere a la alternativa j, aunque la supere en el resto de criterios.

Estos tres umbrales son valores no negativos y además ordenados ($p, q, v \geq 0$; $q \leq p \leq v$).

2.4.4 Cálculo de índices y grado de credibilidad en el modelo Electre III

En este caso la comparación no se realiza únicamente a partir de la valoración de cada alternativa respecto a los criterios definidos en el desarrollo del modelo, sino que tiene en cuenta los umbrales de preferencia p, de indiferencia q, y de veto v, lo que supone la introducción de pseudocriterios (un criterio verdadero es un pseudocriterio con umbrales de preferencia p, de indiferencia q y de veto v, iguales a cero). Por lo tanto, con la introducción de estos umbrales [García L. y Muñoz A., 2009] las preferencias están dadas como se muestra en las siguientes expresiones:

Si $-q_i \leq E_{ij} - E_{ik} \leq q_i$ $\Leftrightarrow$ a_j es indiferente con a_k.

Si $q_i < (E_{ij} - E_{ik}) \leq p_i$ $\Leftrightarrow$ a_j es preferido de forma débil sobre a_k.

Si $p_i < (E_{ij} - E_{ik}) \leq v_i$ $\Leftrightarrow$ a_j es preferido fuertemente a a_k.

Si $(E_{ij} - E_{ik}) \geq v_i$ $\Leftrightarrow$ la posibilidad de superación de a_k sobre a_j es vetada.

Si $-p_i < (E_{ij} - E_{ik}) \leq -q_i$ $\Leftrightarrow$ a_k es preferido de forma débil sobre a_j.

Si $-v_i < (E_{ij} - E_{ik}) \leq -p_i$ $\Leftrightarrow$ a_k es preferido fuertemente a a_j.

Si $(E_{ij} - E_{ik}) \geq -v_i$ $\Leftrightarrow$ la posibilidad de superación de a_j sobre a_k es vetada.

Estas expresiones corresponden a un criterio de maximización; si el criterio es a minimizar se debe invertir el orden de la diferencia.

Electre III utiliza también dos índices: el índice de concordancia y el de discordancia. A diferencia del método Electre, el índice de concordancia se calcula obteniendo primero un índice de concordancia por criterio y luego un índice de concordancia global. El índice de concordancia por criterio afirma en qué medida la alternativa a_j es al menos tan buena como la alternativa a_k para el factor i. Se denota por $c_i(a_j, a_k)$ y se define como:

$$\begin{aligned} c_i(a_j, a_k) = 1 &\Leftrightarrow g_j(E_{ik} - E_{ij}) \leq q_i \\ 0 < c_i(a_j, a_k) < 1 &\Leftrightarrow q_i < (E_{ik} - E_{ij}) \leq p_j \\ c(a_j, a_k) = 0 &\Leftrightarrow p_i < (E_{ik} - E_{ij}) \end{aligned} \tag{2.11}$$

El valor final del índice se calcula mediante interpolación lineal.

El índice de concordancia global se obtiene a partir de la siguiente expresión:

$$IC_{jk} = \frac{\sum_{i=1}^{m} w_i * c_i(a_j, a_k)}{\sum_{i=1}^{m} w_i} \tag{2.12}$$

En donde c_i es el índice de concordancia en el criterio i entre las soluciones a_j y a_k y:

$$c_i(a_j, a_k) \begin{cases} w_i & \text{si } a_j P a_k \\ \frac{1}{2} w_i & \text{si } a_j I_i a_k \\ 0 & \text{en otro caso} \end{cases} \tag{2.13}$$

Por otro lado el índice de discordancia para cada criterio $i[d_i\ (a_j, a_k)]$ se define como:

$$\begin{gathered} d_i(a_j, a_k) = 1 \Leftrightarrow v_i < E_{ik} - E_{ij} \\ 0 < d_i(a_j, a_k) < 1 \Leftrightarrow p_i < (E_{ik} - E_{ij}) \leq v \\ d_i(a_j, a_k) = 0 \Leftrightarrow (E_{ik} - E_{ij}) < p_i \end{gathered} \tag{2.14}$$

Para cada par de alternativas de (a_j, a_k) ahora existe una medida de concordancia y otra de discordancia. En la etapa final de los cálculos se combinan ambas medidas para generar una medida del grado de superación, esto es, una matriz de credibilidad que valora la fortaleza de la afirmación "la alternativa a_j supera a la alternativa a_k". Previamente hay que definir, para cada par de alternativas, el subconjunto $\bar{F}$ de criterios que tiene como elementos aquéllos para los cuales el índice de discordancia por criterio es superior al índice de concordancia global:

$$\bar{F} = \left\{ \frac{i}{i} \in F, d_i(a_j, a_k) > C_{jk} \right\} \tag{2.15}$$

El grado de credibilidad (δ_{jk}) para cada par de alternativas (a_j, a_k) se define como:

$$\delta_{jk} = \begin{cases} C_{jk} & \text{si } \bar{F} = \text{conjunto vacío} \\ C_{jk} * \prod_{i \in \bar{F}} \frac{1 - d_i(a_j, a_k)}{1 - C_{jk}} & \text{en otro casos} \end{cases} \tag{2.16}$$

De la definición del grado de credibilidad se obtiene que éste será el mismo que el valor del índice de concordancia global si ningún valor de los índices de discordancia por criterio es mayor que él. Si para algún criterio ese valor del índice de discordancia por criterio es mayor que el índice de concordancia global, el grado de credibilidad será el índice de concordancia global disminuido por la discordancia. De igual forma, el grado de credibilidad de cualquier par de alternativas será 0 si, para algún criterio, el valor de la discordancia fuera 1.

El siguiente paso en Electre III consiste en ordenar las alternativas a partir de los resultados obtenidos para el grado de credibilidad, llevando a cabo un proceso de destilación descendente y otro ascendente. De ambos procesos se obtienen dos ordenaciones intermedias, que finalmente se aúnan en una ordenación final.

Este método proporciona, por lo tanto, una ordenación final de las alternativas que es una combinación de las obtenidas mediante las destilaciones ascendente y descendente, de tal forma que cuando una alternativa supera a otras, así lo refleja la ordenación final, mientras que si entre las dos ordenaciones hay discrepancia respecto a qué alternativa supera a otra, la ordenación final las considera “incomparables”. Este hecho provoca que la ordenación final sea parcial.

2.5 Sistema relacional de preferencias

Preferencia es un concepto el cual permite elegir entre varios objetos y proporciona un rango de ordenamiento de ellos, basado en la satisfacción que proporcionan al usuario. Por lo tanto la representación más simple de las preferencias del usuario es la clasificación u ordenamiento de objetos [Mobasher B. et. al, 2004].

Los sistemas de preferencias son modelos para formalizar y tratar información sobre preferencias. La forma más directa de mostrar las preferencias es a través de comparaciones por pares. Los sistemas relacionales de preferencias son sistemas de preferencias basados en relaciones binarias. Las relaciones binarias son las herramientas matemáticas más apropiadas para modelar preferencias basadas en comparaciones entre pares de alternativas. El modelo clásico es el sistema relacional simple de preferencias consistente en tres relaciones binarias: la preferencia, la indiferencia y la incomparabilidad. La preferencia y la indiferencia están

presentes en cualquier modelo de preferencias. La incomparabilidad corresponde a la inexistencia de información suficiente para relacionar las alternativas. Un sistema de preferencias se dice completo si no se contempla la incomparabilidad.

Una manera racional de adoptar un sistema de preferencias es usando valoraciones sobre las alternativas y determinando umbrales para las preferencias. Un soporte de valor umbral de un sistema de preferencia es la herramienta formal constituida por valoraciones y umbrales que establecen las relaciones de preferencia e indiferencia del sistema. La preferencia se establece cuando la valoración de una alternativa sobrepasa el umbral de la otra. En los modelos difusos tanto las valoraciones como los umbrales vienen expresados en términos de apreciaciones difusas de las valoraciones y de las alternativas.

2.5.1 Sistema relacional de preferencias difusas del Dr. E. Fernández

Este sistema relacional de preferencias difusas se alimenta de la salida de un método de sobreclasificación difusa, en este trabajo se empleó el método Electre III, es decir requiere como datos de entrada una tabla con los valores de credibilidad (σ) [Roy B., 1990, Roy B.; Slowinski, 2008], utilizando estos valores de credibilidad y los parámetros de simetría (ε), asimetría (β) y el umbral de credibilidad (λ), proporcionados por el DM de acuerdo a sus preferencias para obtener un ordenamiento de las soluciones apegadas a los criterios del DM. Este ordenamiento permite reducir el número de soluciones óptimas proporcionando una mayor calidad en los resultados, reduciendo el número de soluciones no superadas para facilitar la selección de la solución del DM que satisfaga sus preferencias.

Con base en los parámetros σ, ε, β y λ, proporcionados por el DM [Fernández E. et. al, 2013], se genera la tabla de preferencias difusas. Esta tabla contiene el tipo de preferencia de acuerdo a los valores límite ε, β y λ. En la Tabla 2.1 se muestran las categorías que se definen en este modelo.

Tabla 2.1. Categorías de preferencias.

Indiferencia xIy: 1. $\boldsymbol{\sigma(x,y) \geq \lambda \wedge \sigma(y,x) \geq \lambda}$ 2. $\boldsymbol{\lvert\sigma(x,y) - \sigma(y,x)\rvert \leq \epsilon}$	**Incomparabilidad xRy** 1. $\boldsymbol{\sigma(x,y) < 0.5}$ 2. $\boldsymbol{\sigma(y,x) < 0.5}$
Preferencia débil xQy: 1. $\boldsymbol{\sigma(x,y) \geq \lambda \wedge \sigma(x,y) \geq \sigma(y,x)}$ 2. $\boldsymbol{x \neg P(\lambda,\beta) y}$ 3. $\boldsymbol{x \neg I(\lambda,\epsilon) y}$	**Preferencia estricta xPy:** 1. *x domina a y* 2. $\sigma(x,y) \geq \lambda \wedge \sigma(y,x) < 0.5$ 3.$\sigma(x,y) \geq \lambda \wedge (0.5 \leq \sigma(y,x) \leq \lambda) \wedge (\sigma(x,y) - \sigma(y,x)) \geq \beta$
k-Preferencia xKy: 1. $\boldsymbol{0.5 \leq \sigma(x,y) < \lambda}$ 2. $\boldsymbol{\sigma(y,x) < 0.5}$ 3. $\boldsymbol{(\sigma(x,y) - \sigma(y,x)) > \beta/2}$	**No preferencia x~y**

Por otro lado se debe crear la tabla de flujo neto (tabla SWF) en donde cada celda se calcula de la siguiente manera:

$$F_n(a) = \sum_{c \epsilon C - \{a\}} [\sigma(a,c) - \sigma(c,a)] \tag{2.17}$$

La tabla SWF contiene en la primera columna el número de soluciones que tienen una preferencia estricta sobre cada solución ($\#S$), en la segunda columna, contiene el número de soluciones que poseen una preferencia débil sobre cada solución ($\#W$), $\#S$ y $\#W$ se obtienen mediante la tabla de preferencias difusas. Por otro lado, la tercera columna ($\#F$) contiene el número de soluciones que superan a cada resultado (entiéndase de $F_n(a) > F_n(b)$, que $F_n(a)$ supera a $F_n(b)$).

Aquellas soluciones con el menor valor en la columna $\#S$ tienen una mayor preferencia, y en caso de haber más de una, se aplica como criterio de desempate los valores de las otras dos columnas. Ambas tienen la misma jerarquía, y los menores valores son preferidos. Si aún existe un empate, se consideran dichas soluciones con el mismo nivel de preferencia de acuerdo al DM.

2.6 Definición de Heurística

”Una heurística es una técnica que busca soluciones buenas (es decir, casi óptimas) a un costo computacional razonable, aunque sin garantizar factibilidad y optimalidad de las mismas. En algunos casos, ni siquiera puede determinar qué tan cerca del óptimo se encuentra una solución factible en particular” [Coello C., 2013].

2.7 Definición de Metaheurística

El término metaheurística apareció por primera vez en el artículo seminal sobre búsqueda tabú de Fred Glover en 1986. A partir de entonces han surgido multitud de propuestas de pautas para diseñar buenos procedimientos para resolver ciertos problemas que, al ampliar su campo de aplicación, han adoptado la denominación de metaheurísticas [Melián B. et. al, 2003].

“Las metaheurísticas son métodos que integran de diversas maneras, procedimientos de mejora local y estrategias de alto nivel para crear un proceso capaz de escapar de óptimos locales y realizar una exploración robusta en el espacio de búsqueda” [Lazarini L. y López J., 2009].

2.8 Complejidad del problema de Selección de Cartera de Proyectos

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 P 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 $p_1, p_2, \dots, p_n$ 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 $p_1, p_2, \dots, p_n$, 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 R, 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.

   $x[i]$ 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

   $x[i]$:= genera_aleatoriamente_componente(i);

   ENDFOR

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$(x[i] == 1)\{$
   $presupuesto \mathrel{+}= proyectos[i][0]$;
   }
   //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. et. al, 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 2.18) sin exceder la capacidad de carga C (ecuación 2.19).

$$\max(Q_1, Q_2, \dots, Q_n) \tag{2.18}$$

$$Sujeto\ a:$$

$$\sum_{i=1}^{k} w_i\, m_i \leq C \tag{2.19}$$

$$En\ donde:$$

$$Q_i = \sum_{j=1}^{k} b_{i,j}\, m_j \tag{2.20}$$

$$Siendo:$$

Q_i en la ecuación 2.18 representa los objetivos a maximizar, cada uno de los cuales representa el beneficio total de una mochila distinta. w_i en la ecuación 2.19 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 2.20 $b_{i,j}$ 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 2.21) sin exceder el presupuesto total T (ecuación 2.22).

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.

$$\max(O_1, O_2, \dots, O_n) \tag{2.21}$$

$$Sujeto\ a:$$

$$\sum_{i=1}^{k} s_i\, x_i \leq T \tag{2.22}$$

$$En\ donde:$$

$$O_i = \sum_{j=1}^{k} z_{i,j}\, x_j \tag{2.23}$$

$$Siendo:$$

O_i en la ecuación 2.21 los objetivos a maximizar, s_i en la ecuación 2.22 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 2.23 $z_{i,j}$ 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.

2.9 Indicadores de desempeño

Para evaluar el desempeño de un algoritmo multiobjetivo se tienen en cuenta dos aspectos principalmente: su capacidad para minimizar la distancia entre las soluciones compromiso encontradas y las soluciones del frente óptimo de Pareto (o mejor frente encontrado en aquellos problemas en los que no se dispone de éste) y su aptitud para maximizar la dispersión uniforme de las soluciones sobre el frente [Jain H. y Deb K., 2013]. Se han diseñado diferentes indicadores de desempeño que se enfocan en la medición de estas dos características de los algoritmos. A continuación se detallan los indicadores de desempeño utilizados en este trabajo:

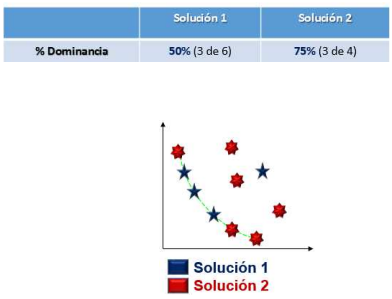


Fig. 2.1. Dominancia

Dominancia. Este indicador trabaja con dos conjuntos de soluciones y aplica el método Non-Fast Dominated Sorting a la unión de ambos, con el fin de dividir las soluciones en frentes y determinar cuántas soluciones de cada conjunto inicial se encuentran en el frente cero (f_0). El porcentaje de dominancia del conjunto uno (c_1) se obtiene considerando el porcentaje que representa el número de soluciones en f_0 que pertenecen a c_1, con relación al total de soluciones de dicho conjunto $\left(\%\ de\ dominancia\ c_1 = \frac{número\ de\ sol.de\ c_1 \in f_0}{número\ de\ sol.\ de\ c_1} * 100\right)$, y de igual forma se realiza este cálculo para el segundo conjunto. Así, cuanto mayor porcentaje de dominancia tenga un conjunto indicará mejor desempeño.

La dispersión (spread), mide el grado de homogeneidad que tienen en su distribución a lo largo del frente, las soluciones obtenidas por el algoritmo analizado. Esta métrica es calculada mediante la ecuación propuesta en [Jain H. y Deb K., 2013], en donde, en vez de utilizar la distancia entre dos soluciones adyacentes, se utiliza la distancia de cada solución a su vecino más cercano:

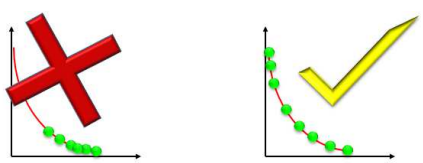


Fig. 2.2. Dispersión (Spread)

$$\Delta = \frac{\sum_{i=1}^{m} d(e_i, S) + \sum_{X \in S} \left| d(X,S) - \bar{d} \right|}{\sum_{i=1}^{m} d(e_i, S) + |S| * \bar{d}} \tag{2.24}$$

En donde S es el conjunto de soluciones, S^* es el conjunto óptimo de Pareto, $(e_1, \dots, e_m)$ son las m soluciones extremas de S^*, m es el número de objetivos y:

$$d(X,S) = \min_{Y \in S, Y \neq X} \|F(X) - F(Y)\|^2, \tag{2.25}$$

$$\bar{d} = \frac{1}{S^*} \sum_{X \in S^*} d(X,S)$$

Esta métrica utiliza las soluciones normalizadas. En el mejor de los casos $\Delta(X,S) = 0$.

El hipervolumen, calcula el volumen en el espacio de objetivos cubierto por un conjunto de soluciones no dominadas o frente de Pareto encontrado (FP_e). Para cada solución $s_i \in FP_e$ se construye un hipercubo h_i a partir de la diagonal definida por un punto de referencia w y la solución s_i. El punto w representa el antióptimo y se obtiene utilizando los peores valores de las funciones objetivo. La unión de todos los hipercubos, considerando el principio de inclusión-exclusión, define el hipervolumen para el frente de soluciones considerado [Jain H. y Deb K., 2013].

$$HV = \text{volume}\left(\bigcup_{i=1}^{|FP_e|} h_i \right) \tag{2.26}$$

La distancia generacional, mide la distancia entre el conjunto de soluciones no dominadas encontradas y el conjunto de óptimos de Pareto [Jain H. y Deb K., 2013].

$$GD = \frac{\sqrt{\sum_{i=1}^{n} d_i^2}}{n} \tag{2.27}$$

En donde:

n es el número de elementos del conjunto de soluciones no dominadas.

d_i es la distancia euclidiana entre cada solución no dominada y el miembro más cercano en el frente de Pareto óptimo.

Del mismo modo que para la métrica Spread, este método requiere que se realice una normalización previa a las soluciones del frente de Pareto encontrado y al frente de Pareto óptimo.

La distancia generacional invertida (IGD: Inverted Generational Distance), consiste en determinar el promedio de las distancias entre cada punto del frente de Pareto verdadero a la solución más cercana del frente de Pareto encontrado. Este indicador mide la proximidad que existe entre el frente de Pareto verdadero (FP_v) y el frente de Pareto encontrado. La expresión matemática para este cálculo es la siguiente [Öztürk M. et. al, 2005]:

$$IGD \triangleq \frac{\left(\sum_{i=1}^{n} d_i^p\right)^{\frac{1}{p}}}{n} \tag{2.28}$$

En donde n es el número de soluciones en el FP_v, p es 2, y d_i es la distancia Euclidiana en el espacio de los objetivos entre cada vector de FP_v y el punto más cercano del FP_e.

Un resultado de 0 indica que ambos frentes son iguales; cualquier otro valor indica una divergencia entre ambos frentes; por ello es deseable obtener valores bajos de IGD.

CAPÍTULO 3

Estado del arte

En el presente capítulo se lleva a cabo una breve descripción de algunas metaheurísticas con las que se ha buscado dar la mejor solución para el problema de Selección de Cartera de Proyectos.

Algoritmos evolutivos multiobjetivo para selección de cartera de inversiones con restricciones de cardinalidad [Colomine F. et. al, 2012]**.** Trabajo publicado en 2009 por Feijoo Colomine, Carlos Cotta y Antonio Fernández. Este trabajo habla de los Algoritmos Evolutivos y se enfoca en la experimentación de tres de ellos para dar solución al problema de Selección de Cartera de Proyectos con restricciones de cardinalidad.

- La primer metaheurística propuesta es Strength Pareto Evolutionary Algorithm 2 (SPEA2), usa una estrategia de grano fino para asignar aptitud, la cual toma en cuenta la cantidad de individuos que cada solución domina y la cantidad de individuos que las dominan, usa una técnica de estimación de densidad de vecinos que guía la búsqueda de manera más eficiente y tiene un esquema de truncamiento de archivo que garantiza la preservación de soluciones de frontera. Usa una selección de Torneo y Elitista.
- El segundo método de solución propuesto es el Nondominated Sorting Genetic Algorithm II (NOSGA-II) que incorpora elitismo y reduce la complejidad del procedimiento de ordenamiento rápido por no dominancia de su antecesor. Realiza una clasificación de la población por frentes. Entre las principales características de esta metaheurística están que mantiene un ordenamiento no dominado elitista que permite que los individuos con mejor aptitud se conserven durante el proceso evolutivo sin

mutarse ni recombinarse, además de permitir la preservación de la diversidad mediante la técnica de Crowding que no necesita de la especificación de parámetros adicionales.

- Finalmente proponen una solución a través del Indicator-Based Evolutionary Algorithm (IBEA), el cual incorpora algunos mecanismos adicionales para mejorar los procesos de búsqueda de óptimos locales y globales en los frentes de Pareto. Emplea indicadores de calidad multiobjetivo para guiar la búsqueda, por lo que no se basa directamente en la noción de dominancia de Pareto. Incorpora prácticas para la toma de decisiones y de información privilegiada para el desarrollo de la búsqueda de la frontera de soluciones de Pareto. Provee de un elemento adicional que es el de la utilización de medidas de desempeño (indicadores binarios de calidad) que permite generar una muy buena y rápida respuesta.

Preference Incorporation into evolutionary multiobjective optimization using preference information implicit in a set of assigment examples [Cruz-Reyes L. et al., 2013a]. Texto escrito en 2013 por Laura Cruz Reyes, Eduardo Fernández y Patricia Sánchez. Plantea el uso de métodos de clasificación multicriterio combinado con un algoritmo multiobjetivo incorporando preferencias con el fin de que una vez encontrado el frente de Pareto se pueda obtener la mejor solución compromiso dentro de la región de interés.

El algoritmo propuesto es Hybrid Multi-Criteria Sorting Genetic Algorithm (H-MCSGA, Algoritmo Genético Híbrido de Clasificación Multicriterio), el cual consiste en dos fases:

1) Mediante un enfoque metaheurístico multiobjetivo se obtiene una aproximación a la frontera de Pareto. El DM toma esa aproximación.
2) Clasifica en un conjunto de categorías ordenadas para construir un conjunto referencia al que se le aplica el método de clasificación multicriterio Theseus. En esta segunda fase la información de las buenas soluciones es usada por una variante del popular NSGA-II para guiar la búsqueda hacia la Región de Interés (ROI).

Las preferencias del DM pueden ser expresadas de diversas formas: Pesos, clasificación de soluciones, clasificación de objetivo, punto de referencia, punto de reservación, objetivos entre intercambios, umbrales de conveniencia y parámetros de sobreclasificación.

Un algoritmo multiobjetivo basado en Búsqueda Dispersa [Nebro A. et. al, 2007]. Artículo publicado en el año 2006, este trabajo ofrece una solución a los problemas multiobjetivo de la familia ZDT mediante un híbrido de la metaheurística SS al que llaman Archive-based hybrid scatter search debido a que basa su funcionamiento en la metaheurística SS a la que añade un método de cruza Simulated Binary Crossover (SBX) y uno de mutación (1+1) EA, además de la particularidad del uso de un archivo histórico que almacena aquellas soluciones que han sido descartadas a través de las generaciones como lo hace PAES.

Como indicadores de calidad emplea tres técnicas: Distancia Generacional, Hipervolumen y una variante de Spread.

A steady state decomposition based quantum genetic algorithm for many objective optimization [Tepabrata R. et. al, 2013]. Trabajo que se publicó en 2013, plantea una solución a problemas multiobjetivos a través de un algoritmo genético cuántico, proporciona los resultados obtenidos al aplicar esta metaheurística al problema ZDT2.

Aplica el método de Hipervolumen como indicador de calidad.

An improved adaptive approach for elitist nondominated sorting genetic algorithm for many-objective optimization [Jain H. y Deb K., 2013]. En este trabajo realizado en 2013 se observa el desempeño de las metaheurísticas NSGA-III, A-NSGA-III y A^2-SGA-III sobre los problemas de la familia ZDT, mostrando las diferencias en los resultados de cada una de estas técnicas y planteando en qué difieren.

Utiliza el indicador Hipervolumen para medir la calidad de cada uno de los métodos empleados.

An evolutionary many-objective optimization algorithm using reference-point based non-dominated sorting approach, part I: solving problems with box constraints [Deb K. y Jain H., 2014]. Primer trabajo que forma parte de una serie que habla de la evolución de la metaheurística NSGA-II y que muestran cómo se van dando los cambios en ella pasando a ser MO-NSGA-II (mejor conocido como NSGA-III) y su desempeño sobre problemas de muchos objetivos como los pertenecientes a la familia ZDT (en los trabajos posteriores se explica cómo prosigue la evolución a A-NSGA-III y A^2-NSGA-III).

Como indicadores de calidad emplea las técnicas MOEA/D-PBI, que es una medida de distancia penalizada de un punto, el punto ideal está formado por la suma ponderada de la dirección de referencia y la distancia a lo largo de la dirección de referencia. La segunda técnica es la métrica Tchebycheff usando un punto ideal y un vector de pesos, a esta técnica adaptada para las variantes de NSGA-III y sus puntos de referencia se le llama MOEA/D-TCH.

Multicriteria optimization of interdependent project portfolio with "a priori" incorporation of decisión maker preferences [Cruz-Reyes L. et al., 2013b]**.** Artículo escrito en el año 2014 por Laura Cruz Reyes, Eduardo R. Fernández, Claudia G. Gómez y Gilberto Rivera. Proponen una solución al problema de Cartera de Proyectos con interdependencia mediante una metaheurística llamada Non-Outranked Ant Colony Optimization (NO-ACO), tal como su nombre lo indica tiene su base en el algoritmo ACO, sin embargo incorpora la utilización de preferencias a priori tomando en cuenta las preferencias de un tomador de decisiones, mediante la utilización del modelo de sobreclasificación descrito en la sección 2.5.1.

NO-ACO toma en cuenta sinergia y redundancia durante la formación de la cartera.

Afirma que a través de la incorporación de preferencias, la presión selectiva hacia una zona privilegiada de la frontera de Pareto se incrementa y con ello la zona que coincide con las preferencias del DM puede ser identificada.

Abre la posibilidad de incorporar en trabajos futuros el manejo de apoyo parcial.

En la Tabla 3.1 se muestra un resumen de las características relevantes de cada trabajo revisado.

Tabla 3.1. Trabajos relacionados.

Autores	Metaheurística	Indicadores de calidad	Instancias
Feijoo Durán, Carlos Cotta y Antonio J. Fdez. (2009)	• NSGA II. • SPEA 2. • IBEA.	• Hipervolumen. • R2.	Valores reales de la bolsa de valores de Caracas.
Patricia Sánchez. (2014)	• H-MCSGA.	• Dominancia.	Instancias para el problema de la Cartera de Proyectos creadas por un generador de instancias.
Antonio Nebro, Francisco Luna, Enrique Alba, Bernabé Dorronsoro y Juan J. Durillo. (2006)	• ABYSS (Archive-Based hYbrid Scatter Search).	• Distancia Generacional. • Variante de Spread. • Hipervolumen.	Instancias de los problemas: Shcaffer, Fonseca, Kursawe, ZDT1, ZDT2, ZDT3, ZDT4, ZDT5, ZDT6, ConstrEx, Srinivas, Osyczka2, Golinski, Tanaka, Viennet2, Viennet3, Viennet4 y Water.
Ray Tepabrata, Md Asafuddoula y Amitay Isaacs. (2013)	• Algoritmo genético cuántico.	• Hipervolumen.	Instancia del problema estándar DTLZ2.
Himanshu Jain y Kalyanmoy Deb. (2013)	• NSGA-III. • A-NSGA-III. • A^2-NSGA-III.	• Hipervolumen.	Instancias de DTLZ1 y DTLZ2 invertidos, DTLZ1 y DTLZ2 con una restricción, Crashworthiness, Car Side Impact, Machining, Water.
Kalyanmoy Deb y Himanshu Jain. (2009)	• NSGA-III.	• MOEA/D-PBI. • MOEA/D-TCH.	Instancias de los problemas estándar DTLZ1, DTLZ2, DTLZ3, DTLZ4, WFG6 y WFG7.
Laura Cruz, Eduardo Fdz., Claudia Gómez S. y Gilberto Rivera. (2014)	• NO-ACO	• Dominancia.	Instancias para el problema de la Cartera de Proyectos creadas por un generador de instancias.

CAPÍTULO 4

Metaheurísticas multiobjetivo

Una de las clasificaciones para las metaheurísticas está dada por el número de soluciones que procesan en un determinado momento durante el proceso de búsqueda, es decir, aquellas "basadas en trayectoria" y las "basadas en población" o poblacionales. En el caso de las poblacionales trabajan con un conjunto de soluciones que iterativamente son mejoradas a través de un proceso inteligente de exploración del espacio de búsqueda [Lazarini L. y López J., 2009].

Dentro del conjunto de metaheurísticas poblacionales, las más difundidas corresponden a las técnicas basadas en la Computación Evolutiva: Algoritmos Genéticos (AGs), Estrategias Evolutivas (EEs), Programación Evolutiva (PE) y Programación Genética (PG). Todos estos, generalmente referidos como Algoritmos Evolutivos (AEs). Existen además otros enfoques alternativos, que si bien presentan algunas de las características de los AEs, tienen rasgos particulares que los diferencian de estos últimos dentro de los que podemos mencionar a la Optimización mediante Cúmulo de Partículas (PSO), la Evolución Diferencial (DE), la Optimización por Colonia de Hormigas (ACO), los Algoritmos de Estimación de Distribución (EDA) o SS. Estas metaheurísticas han mostrado ser sumamente eficientes cuando se aplican a problemas complejos; ya sea de laboratorio o del mundo real. Si bien en ningún caso aseguran la obtención del mínimo o máximo global, han mostrado que son capaces de alcanzar soluciones de calidad en periodos de tiempo aceptables [Lazarini L. y López J., 2009].

Como se planteó en la sección 1.3 el objetivo de este trabajo es diseñar e implementar una metaheurística. En este trabajo se realiza un estudio de las metaheurísticas que han mostrado

un mejor desempeño de acuerdo al estado del arte de problemas multiobjetivo, tales como NSGAII, A-NSGAII y A^2-NSGAIII.

4.1 Introducción a los Algoritmos Genéticos

La capacidad del ser humano para predecir el comportamiento de su entorno, se ha ido incrementando con el paso del tiempo. La inteligencia artificial es responsable de muchos de esos logros. Los pioneros de esta ciencia estaban tan interesados en la electrónica, como en la biología, y por eso sus aplicaciones iban desde calcular trayectorias de misiles, a tratar de modelar el cerebro, de imitar el proceso de aprendizaje humano, y de simular la evolución biológica. La década de los ochenta marcó el inicio de un gran interés de la comunidad científica por estos temas computacionales inspirados en la biología, que han visto como su desarrollo les llevaba a cosas inimaginables, primero en el campo de las redes neuronales, luego en el del aprendizaje, y por último en lo que ahora se conoce como "computación evolutiva", de la que los Algoritmos Genéticos constituyen su máximo exponente [Rodríguez-Piñero T., 2003].

El científico considerado creador de los Algoritmos Genéticos es John Holland, que los desarrolló, junto a sus alumnos y colegas, durante las décadas de 1960 y 1970. En contraste con las estrategias evolutivas y la programación evolutiva, el propósito original de Holland no era diseñar algoritmos para resolver problemas concretos, sino estudiar, de un modo formal, el fenómeno de la adaptación tal y como ocurre en la naturaleza, y desarrollar vías de extrapolar esos mecanismos de adaptación natural a los sistemas computacionales. El Algoritmo Genético de Holland era un método para desplazarse, de una población de cromosomas (bits) a una nueva población, utilizando un sistema similar a la "selección natural" junto con los operadores de cruces, mutaciones e inversión inspirados en la genética. En este primitivo algoritmo, cada cromosoma consta de genes (bits), y cada uno de ellos es una muestra de un alelo particular (0 o 1). El operador de selección escoge, entre los cromosomas de la población, aquellos con capacidad de reproducción, y entre éstos, los que sean más "compatibles", producirán más descendencia que el resto. El de cruce extrae partes de dos cromosomas, imitando la combinación biológica de dos cromosomas aislados (gametos). La mutación se encarga de cambiar, de modo aleatorio, los valores del alelo en

algunas localizaciones del cromosoma; y, por último, la inversión, invierte el orden de una sección contigua del cromosoma, recolocando por tanto el orden en el que se almacenan los genes. La mayor innovación de Holland fue la de introducir un algoritmo basado en poblaciones con cruces, mutaciones e inversiones [Rodríguez-Piñero T., 2003].

Algoritmo Básico

1. Generar (aleatoriamente) una población inicial.
2. Calcular aptitud de cada individuo.
3. Seleccionar (probabilísticamente) con base en aptitud.
4. Aplicar operadores genéticos (cruza y mutación) para generar la siguiente población.
5. Ciclar hasta que cierta condición se satisfaga.

Fig. 4.1 Ciclo de un Algoritmo Genético básico.

1. *Genera población*. Es el primer paso, si es la primera vez que se lleva a cabo funge como el paso creador en la evolución, es decir, crea una especie con N individuos por primera vez con ciertas características, semejantes entre ellos pero sin repeticiones. A partir de la segunda vez que se ejecute hará un filtrado entre la unión de padres e hijos

para saber quiénes sobreviven y forman parte de la población de la siguiente generación.

2. *Selección*. A partir de la población existente de cualquier especie, para poder mantenerse y sobrevivir necesita tener descendencia, al igual que en la vida el proceso de selección emula el hecho de que los más aptos deben ser quienes generen dicha descendencia eligiendo aquellas soluciones (individuos) cuya genética prevalecerá sobre el resto para la siguiente generación.
3. *Cruza*. Este método emula el proceso de reproducción, posee un porcentaje de éxito el cual indica que no todos los seres son capaces de procrear, entiéndase aquellos cuyas características físicas les impiden dicho proceso, debido a que la mayor parte de los individuos tienen éxito en este proceso en un ámbito real, este porcentaje suele ser alto (ejemplo 90%).
4. *Mutación*. Ayuda a la exploración del espacio de búsqueda mediante la diversificación de las soluciones, emula el hecho de que los hijos no poseen 100% características de los padres, sino que de vez en cuando surgen cambios mínimos que le brindan atributos que no posee ni la madre ni el padre pero que genéticamente puede llevar arrastrando, en el ámbito real se hablaría de genes recesivos como el color de ojos, piel, tipo de cabello, etc. Debido a que son cambios mínimos el porcentaje de éxito de este método es pequeño, dependiendo del tipo de proceso empleado suele ir de 1% al 10% generalmente, no se recomienda un gran porcentaje de mutación ya que si los cambios son muchos la solución sería equivalente a una generada al azar y se perderían las buenas características arrastradas durante las generaciones anteriores.
5. *Unión de padres e hijos*. Las soluciones factibles surgidas en la presente generación se une a la población anterior siempre y cuando no se encuentren ya dentro de ella. Se hace un filtrado en el cual solo N cantidad de individuos establecida en un principio permanecen dentro de la población, el resto de los individuos perece o queda fuera antes de proceder a recomenzar el ciclo para la siguiente generación.

Este ciclo se repite M cantidad de veces preestablecidas.

Elementos de un Algoritmo Genético

La representación tradicional de una solución es la cadena binaria del tipo:

0110	1101	0011	1001
Cadena 1	*Cadena 2*	*Cadena 3*	*Cadena4*

A la cadena se le llama “cromosoma”. A cada posición de la cadena se le denomina “gen” y el valor dentro de esta posición se le llama “alelo”.

Para poder aplicar el Algoritmo Genético se requiere de los siguientes componentes básicos:

1. Una representación de las soluciones del problema.
2. Una forma de generar una población inicial de posibles soluciones.
3. Una función de evaluación que clasifique las soluciones en términos de aptitud.
4. Operadores genéticos que alteren la composición de los hijos que se producirán para las siguientes generaciones.
5. Valores para los diferentes parámetros que utiliza el Algoritmo Genético (tamaño de la población, probabilidad de cruza, probabilidad de mutación, número máximo de generaciones, etc.)

4.2 Non-dominated Sorting Genetic Algorithm II (NSGA-II)

Durante la década de los 90’s fue propuesto un conjunto de Algoritmos Evolutivos multiobjetivo, la principal razón fue su habilidad para encontrar múltiples soluciones Pareto óptimas en una sola corrida. Entre dichos algoritmos se encontraba el Non-dominated Sorting Genetic Algorithm (NSGA) [Srinivas N. y Deb K., 1994].

El algoritmo NSGA-II (Non-dominated Sorting Genetic Algorithm, versión II) fue presentado por K. Deb en el Instituto Tecnológico Kanpur en India en el año 2000 [Deb et al., 2000]. Surgió como una versión mejorada del algoritmo NSGA [Srinivas N. y Deb K., 1994], de quién heredó su estructura principal, pero incorpora o incluye características distintivas para resolver los problemas de los altos costos computacionales involucrados con el ordenamiento por no dominación, la dificultad que conlleva la necesidad de especificar un parámetro para asegurar la diversidad en la población y la lenta convergencia al frente óptimo

debido a la ausencia de elitismo. Las características principales del algoritmo NSGA-II abarcan:

- Un enfoque de ordenamiento rápido por no-dominancia que utiliza una técnica de comparación apoyado en una subpoblación auxiliar, el cual permite disminuir la complejidad en la verificación de dominancia de $O(MP^3)$ a $O(MP^2)$, siendo M el número de funciones objetivo y P el tamaño de la población utilizada [Nesmachnow S., 2004].
- La utilización de la técnica de Crowding que no requiere especificar parámetros adicionales para la preservación de diversidad en la población, eliminando la dependencia del parámetro de compartición σ utilizado por el NSGA original [Nesmachnow S., 2004].

4.2.1 Descripción de la metaheurística NSGA-II

NSGA-II basa su estrategia en la combinación de las soluciones de mejor calidad elegidas a través de la agrupación del conjunto solución en frentes, aplicándoles posteriormente un método de selección por Torneo Binario y la incorporación de soluciones que brindan diversidad, las cuales son seleccionadas mediante el método de Distancia de Crowding. NSGA-II está conformado por 6 métodos principales:

1. **Generación de la población.** Genera aleatoriamente la población inicial de padres con un tamaño N predefinido, comprueba la factibilidad y no repetición de cada miembro de la población.
2. **Clasificación por no dominancia.** Este método clasifica en frentes de Pareto a toda la población actual.
3. **Selección.** Utiliza una estrategia por Torneo Binario, en la que se elige de la población de padres (tomados de dos en dos) y de los cuales aquel que pertenece al mayor frente es agregado a la lista de padres que generarán a los hijos de esta generación, en caso de pertenecer al mismo frente se elige uno al azar.
4. **Cruza.** Genera los miembros de la siguiente generación (dos por cada par de soluciones elegidas en el método de selección), se utiliza una cruza tipo SBX cuando se trata de números reales y cruza de un punto cuando se trabaja con soluciones binarias. En la

cruza en Un Punto se toman dos soluciones padre como entrada, se genera un número aleatorio que indica el punto de corte de las soluciones padre, de esta forma la solución $hijo_1$ estará conformada por los n primeros genes del $padre_1$ (desde el gen cero hasta el punto de corte) y los m genes restantes se toman del $padre_2$ (desde el punto de corte hasta el último gen). Para generar el $hijo_2$ se aplica el mismo procedimiento invirtiendo el orden de selección de los padres [Deb K. et. al, 2002].

5. **Mutación**. Esta metaheurística trabaja con una estrategia de Mutación Binaria que consiste en recorrer la solución gen por gen y utiliza un número aleatorio cuyo rango es 0-1 (probabilidad de mutación) para determinar si aplica un inversión del valor binario contenido en el gen. Para el caso de números reales este algoritmo utiliza un método de Mutación Polinomial [Deb K. et. al, 2002], de acuerdo a su autor este método emula las operaciones de la Mutación Binaria con números reales al no ser posible el aplicar ese método por las características del tipo de datos. La Mutación Polinomial genera soluciones cercanas a la solución que se está mutando con una probabilidad más alta que soluciones lejanas a ella, utiliza una distribución de probabilidad polinomial por lo cual este método recibe ese nombre, se ha empleado en metaheurísticas como SPEA2 [Zitzler E. et. al, 2001] además de NSGA-II.
6. **Distancia de Crowding**. Método utilizado para proporcionar diversidad a las soluciones explorando otros lugares del espacio de búsqueda que de otra manera serían omitidos al caer en óptimos locales. Este procedimiento se describe en la siguiente sección.

4.2.2 Estrategias de diversidad de la metaheurística NSGA-II

La Distancia de Crowding es una técnica utilizada para proporcionar diversidad a las soluciones en la metaheurística NSGA-II y evitar eventos como el caer en óptimos locales, permitiendo la posible exploración de zonas en el espacio de búsqueda que posiblemente sin su aplicación (o el de otro método diversificador) no se darían. A continuación se muestra el algoritmo para este proceso [Deb K. et. al, 2000].

DISTANCIA DE CROWDING

1. $l = |I|$ //Número de soluciones en I
2. **Para cada** i
3. $I[i]distancia = 0$ //Inicializa distancias
4. **Para cada** objetivo m
5. $I = Ordena(I, m)$ //Ordena utilizando cada valor objetivo
6. $I[0]_{distancia} = I[l]_{distancia} = \infty$ //Los puntos extremos son siempre seleccionados
7. **Para** $i = 2$ hasta $(l - 1)$
8. $I[i]_{distancia} += \frac{I[i+1].m - I[i-1].m}{f_m^{max} - f_m^{min}}$ //Para todos los otros puntos

Algoritmo 4.1 Distancia de Crowding

Lo que busca la Distancia de Crowding es encontrar aquellas soluciones que se encuentren en un espacio menos conglomerado de otras soluciones y les permite pasar a la siguiente generación con el fin de que esa diferencia que las coloca en otra región del espacio de búsqueda permita generar nuevas soluciones con una calidad igual o superior a la de las soluciones que pasaron a dicha generación debido a su calidad pero colocadas en una región del espacio de búsqueda distinto, permitiendo explorar y explotar otras regiones en las siguientes generaciones.

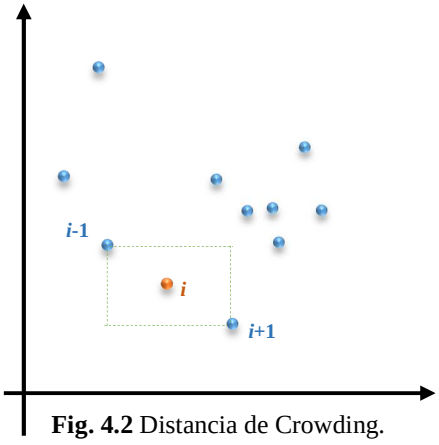


Fig. 4.2 Distancia de Crowding.

4.2.3 Algoritmo NSGA-II

NSGA-II

Entrada: Población de padres P_t [Deb K. et al., 2000].

Salida: P_{t+1}

1. Generar P_t.
2. $F = Clasificación\_por\_no\_dominancia\ (P_t)$
3. $P_t = Selección\_torneo\_binario(P_t)$
4. $Q_t = Cruza\_SBX(P_t)$
5. $Q_t = Mutación\_polinomial(Q_t)$
6. **Mientras** $(\boldsymbol{generaciones < n})$
7. $R_t = P_t \cup Q_t$
8. $P_{t+1} = \emptyset \ e \ \ i = 1$
9. **Hacer**
10. $P_{t+1} = P_{t+1} \cup F_i$
11. $i = i + 1$
12. **Mientras que** $|P_{t+1}| + |F_i| \leq N$
13. **Si** $(|P_{t+1}| \ != N)$
14. *DistanciaCrowding*(F_i)
15. $P_{t+1} = P_{t+1} \cup F_i\ [1: (N - |P_{t+1}|)]$
16. $Q_{t+1} = generaNuevaPoblacionDeHijos(P_{t+1})$
17. $t = t + 1$
18. $F = clasificaciónPorNoDominancia(P_t)$

Algoritmo 4.2 NSGA-II

4.2.4 Evaluación experimental del algoritmo NSGA-II

Las experimentaciones se llevaron a cabo en un ordenador con las siguientes características:

Hardware. Procesador Inside i3 2.1 Ghz de 64 bits, 6 Gb en RAM y HDD 7200 rpm.

Software. S.O. Windows 8 x64, lenguaje Java, JDK 1.8.0, IDE NetBeans 8.0.1.

Se realizó un ajuste de parámetros a las cuatro metaheurísticas que intervinieron en éste, dando como resultado las siguientes configuraciones iniciales:

NSGA-II.- La codificación de este algoritmo se realizó apegado a [Deb K. et al., 2000] y se determinó que la mejor configuración es un tamaño de: 100 para la población inicial. Aplica una probabilidad de cruza de 90%, utiliza un método de cruza en Un Punto. Se llevó a cabo la implementación de una mutación Binaria con una probabilidad de ejecución del 1%, el método de selección por Torneo Binario combinado con la unión de frentes ayudados por Distancia de Crowding y la condición de terminación del algoritmo son 500 generaciones.

NSGA-III y A^2-NSGA-III.- La implementación de estas metaheurísticas se llevó a cabo basándose en [Jain H. y Deb K., 2013], configuración empleada para estas metaheurísticas fue un tamaño de: 105 para la población inicial. Tiene una probabilidad de cruza de 90% utilizando un método de cruza en Un punto, emplea una mutación Binaria con una probabilidad de 1%, el método de selección es una selección aleatoria aplicando diversidad mediante una función de Nicho y la condición de paro del algoritmo se da al cumplirse 500 generaciones.

MSS: Emplea un tamaño de: 100 para el conjunto P, 20 para $RefSet_1$, 10 para $RefSet_2$, 20 para el archivo externo. Por otro lado, aplica una probabilidad de combinación de 100%, utiliza el método de combinación Uniforme (cada 5 iteraciones) para el proceso de combinación de subconjuntos, la mejora implementa el método (1+1) EA y la condición de terminación del algoritmo son 700 mil evaluaciones de la función objetivo.

Para cada uno de los experimentos se realizan 30 ejecuciones de la metaheurística correspondiente y se utilizaron dos instancias pertenecientes al problema de Selección de Cartera de Proyectos: o3p100_1 y o9p100_1, estas instancias trabajan con tres y nueve objetivos respectivamente y ambas contienen 100 proyectos.

4.2.5 Resultados y conclusiones

Tabla 4.1 Resultados de la experimentación para la instancia de tres objetivos mediante el indicador de desempeño de Dominancia.

	Corrida	NSGA-II	NSGA-III	A^2-NSGA-III	MSS
% de soluciones no superadas	1	**88% (88 de 100)**	47% (50 de 105)	41% (44 de 105)	68% (20 de 29)
	2	72% (72 de 100)	42% (45 de 105)	59% (62 de 105)	**89% (26 de 29)**
	3	**76% (76 de 100)**	51% (54 de 105)	71% (75 de 105)	65% (19 de 29)
	4	**90% (90 de 100)**	49% (52 de 105)	49% (52 de 105)	86% (25 de 29)
	5	**72% (72 de 100)**	57% (60 de 105)	50% (53 de 105)	**72% (21 de 29)**
	6	**88% (88 de 100)**	58% (61 de 105)	55% (58 de 105)	72% (21 de 29)
	7	66% (66 de 100)	54% (57 de 105)	42% (45 de 105)	**89% (26 de 29)**
	8	73% (73 de 100)	48% (51 de 105)	62% (66 de 105)	**96% (28 de 29)**
	9	79% (79 de 100)	54% (57 de 105)	37% (39 de 105)	**87% (34 de 39)**
	10	77% (77 de 100)	58% (61 de 105)	50% (53 de 105)	**84% (33 de 39)**
	11	61% (61 de 100)	38% (40 de 105)	30% (32 de 105)	**89% (35 de 39)**
	12	60% (60 de 100)	43% (46 de 105)	47% (50 de 105)	**92% (36 de 39)**
	13	73% (73 de 100)	56% (59 de 105)	42% (45 de 105)	**87% (34 de 39)**
	14	74% (74 de 100)	52% (55 de 105)	**80% (84 de 105)**	74% (29 de 39)
	15	79% (78 de 98)	42% (45 de 105)	84% (89 de 105)	**89% (35 de 39)**
	16	42% (42 de 100)	47% (50 de 105)	**95% (100 de 105)**	79% (31 de 39)
	17	63% (63 de 100)	53% (56 de 105)	**85% (90 de 105)**	35% (14 de 39)
	18	38% (38 de 100)	51% (54 de 105)	59% (62 de 105)	**94% (37 de 39)**
	19	76% (76 de 100)	52% (55 de 105)	**80% (85 de 105)**	56% (22 de 39)
	20	62% (62 de 100)	48% (51 de 105)	82% (87 de 105)	**92% (36 de 39)**
	21	43% (43 de 100)	50% (53 de 105)	69% (73 de 105)	**82% (32 de 39)**
	22	41% (41 de 100)	44% (47 de 105)	**87% (92 de 105)**	79% (31 de 39)
	23	76% (76 de 100)	40% (43 de 105)	**92% (97 de 105)**	87% (34 de 39)
	24	38% (38 de 100)	50% (52 de 102)	68% (72 de 105)	**87% (34 de 39)**
	25	62% (62 de 100)	47% (50 de 105)	76% (80 de 105)	**82% (32 de 39)**
	26	44% (44 de 100)	48% (51 de 105)	83% (88 de 105)	**84% (33 de 39)**
	27	76% (76 de 100)	36% (38 de 105)	73% (77 de 105)	**79% (31 de 39)**
	28	36% (36 de 100)	57% (60 de 105)	**87% (92 de 105)**	82% (32 de 39)
	29	52% (52 de 100)	42% (45 de 105)	76% (80 de 105)	**89% (35 de 39)**
	30	54% (54 de 100)	50% (53 de 105)	66% (70 de 105)	**84% (33 de 39)**

Las celdas sombreadas en color verde representan un resultado que supero al del resto de la metaheurísticas, las celdas color anaranjado simbolizan que además de ellas, en la misma corrida hubo otra metaheurística que presento el mismo desempeño, la cual se encuentra del mismo color en la correspondiente línea.

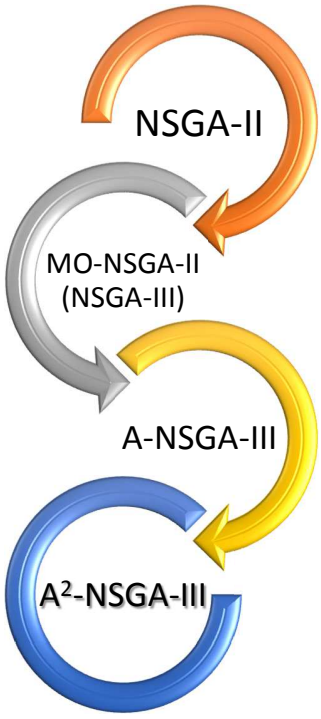
Tabla 4.2 Resultados de la experimentación para la instancia de nueve objetivos mediante el indicador de desempeño de Dominancia.

	Corrida	NSGA-II	NSGA-III	A^2-NSGA-III	MSS
% de soluciones no superadas	1	13% (13 de 100)	19% (32 de 165)	41% (68 de 165)	**92% (36 de 39)**
	2	3% (3 de 100)	30% (51 de 165)	35% (59 de 165)	**94% (37 de 39)**
	3	34% (34 de 100)	50% (83 de 165)	41% (69 de 165)	**94% (37 de 39)**
	4	64% (64 de 100)	25% (42 de 165)	61% (102 de 165)	**97% (38 de 39)**
	5	54% (54 de 100)	54% (54 de 100)	89% (147 de 165)	**92% (36 de 39)**
	6	74% (74 de 100)	55% (92 de 165)	56% (93 de 165)	**79% (31 de 39)**
	7	15% (15 de 100)	24% (41 de 165)	63% (104 de 165)	**92% (36 de 39)**
	8	16% (16 de 100)	54% (90 de 165)	65% (108 de 165)	**94% (37 de 39)**
	9	39% (39 de 100)	36% (61 de 165)	52% (87 de 165)	**92% (36 de 39)**
	10	62% (62 de 100)	76% (127 de 165)	**87% (144 de 165)**	**87% (34 de 39)**
	11	**96% (96 de 100)**	43% (71 de 165)	48% (80 de 165)	94% (37 de 39)
	12	30% (30 de 100)	44% (74 de 165)	82% (136 de 165)	**94% (37 de 39)**
	13	64% (64 de 100)	65% (108 de 165)	69% (115 de 165)	**89% (35 de 39)**
	14	49% (49 de 100)	27% (46 de 165)	55% (91 de 165)	**92% (36 de 39)**
	15	86% (86 de 100)	36% (61 de 165)	61% (101 de 165)	**89% (35 de 39)**
	16	8% (8 de 100)	32% (54 de 165)	23% (38 de 165)	**92% (36 de 39)**
	17	91% (91 de 100)	61% (101 de 165)	75% (125 de 165)	**92% (36 de 39)**
	18	**95% (95 de 100)**	44% (73 de 165)	75% (125 de 165)	92% (36 de 39)
	19	91% (91 de 100)	30% (50 de 165)	41% (68 de 165)	**97% (38 de 39)**
	20	63% (63 de 100)	42% (70 de 165)	66% (109 de 165)	**97% (38 de 39)**
	21	**95% (95 de 100)**	35% (58 de 165)	49% (82 de 165)	87% (34 de 39)
	22	62% (62 de 100)	61% (101 de 165)	72% (120 de 165)	**89% (35 de 39)**
	23	16% (16 de 100)	47% (79 de 165)	55% (91 de 165)	**92% (36 de 39)**
	24	53% (53 de 100)	41% (68 de 165)	82% (136 de 165)	**97% (38 de 39)**
	25	47% (47 de 100)	20% (33 de 165)	51% (85 de 165)	**97% (38 de 39)**
	26	81% (81 de 100)	46% (77 de 165)	31% (52 de 165)	**82% (32 de 39)**
	27	35% (35 de 100)	62% (103 de 165)	66% (109 de 165)	**97% (38 de 39)**
	28	36% (36 de 100)	44% (73 de 165)	83% (137 de 165)	**92% (36 de 39)**
	29	33% (33 de 100)	36% (60 de 165)	86% (143 de 165)	**89% (35 de 39)**
	30	69% (69 de 100)	31% (52 de 165)	47% (79 de 165)	**84% (33 de 39)**

Se concluye que tal como se puede observar en las tablas 4.1 y 4.2, NSGA-II brinda un mejor desempeño a menor número de objetivos que intervengan en el problema, conforme se incrementa la cantidad de objetivos que intervienen en el problema, la calidad de sus

resultados decrece sobre todo en comparación con los resultados de MSS y las pocas corridas en las que demuestra ser mejor lo hace por poco, mientras que en aquellas que es superada lo hace por una desventaja muy marcada a diferencia de las corridas para la instancia de tres objetivos. También es destacar que NSGA-II presenta mejores resultados que sus sucesores NSGA-III y A^2-NSGA-III, para la instancia de tres objetivos no se ve superado en alguna corrida por NSGA-III, aunque A^2-NSGA-III presenta un mejor desempeño. Para la instancia de nueve objetivos NSGA-II de la misma forma presenta mejores resultados que NSGA-III y además supera por un poco los resultados de A^2-NSGA-III.

4.3 NSGA-III, Adaptive NSGA-III (A-NSGA-III) y Efficiently Adaptive NSGA-III (A^2-NSGA-III)



- Metaheurística para resolver problemas multiobjetivo.
- Incorpora el uso de puntos de referencia como estrategia de diversificación.
- Mejora el funcionamiento de NSGA-III a través de la actualización de los puntos de referencia, mediante métodos de adición (simplex *M*-1 dimensional) y eliminación de puntos.
- Mejora el procedimiento de Actualización de Puntos de Referencia utilizando el mismo procedimiento base simplex *M*-1 dimensional pero utilizando el punto que genera el simplex como uno de los puntos a añadir.

Fig. 4.3 Evolución NSGA-II => A^2-NSGA-III

NSGA-III es una metaheurística que ha sido aplicada a problemas de optimización con muchos objetivos. Los conceptos fundamentales y principios del método fueron propuestos por Kalyanmoy Deb y Himanshu Jain a principios de la década pasada [Deb K. y Jain H., 2013].

Esta metaheurística surge de la necesidad de dar solución al problema de Selección de Cartera de Proyectos en un ámbito real, debido a que los métodos de optimización multiobjetivo solo arrojan buenos resultados al trabajar con menos de cinco objetivos, cuando un problema real típicamente incorpora más de diez objetivos.

4.3.1 Estrategias de diversidad de las metaheurísticas NSGA-III, A-NSGA-III y A^2-NSGA-III

NSGA-III se apoya en la metaheurística NSGA-II tomándola como base pero modificando la forma de proporcionar la diversidad a las soluciones, a diferencia de su antecesor, ésta metaheurística en vez de utilizar el método "Distancia de Crowding", utiliza un conjunto de puntos de referencia que guían a las soluciones creadas a través de las generaciones repartiéndolas sobre un hiperplano normalizado que se genera mediante el conjunto de soluciones iniciales, adaptando su forma a la del frente óptimo de Pareto, utilizando para ello un indicador de cercanía de las soluciones a las líneas de referencia que se forman del origen hacia cada uno de los puntos de referencia tendiendo al infinito [Deb K. y Jain H., 2013].

NSGA-III tiene el inconveniente de que no todos los puntos de referencia pueden ser asociados con un conjunto Pareto óptimo bien distribuido, esto conlleva a desperdiciar esfuerzo computacional debido a los cálculos producidos por el conjunto de puntos de referencia que nunca se asociaron y fueron de la primera a la última generación. Una solución a este inconveniente es primero encontrar todos los puntos de referencia que no están asociados con un miembro de la población, entonces en vez de eliminarlos se reposicionan, dicha estrategia fue propuesta en A-NSGA-III [Jain H. y Deb K., 2013].

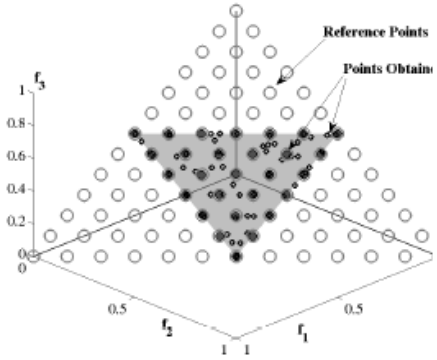


Fig. 4.4 Solo 28 de 91 puntos de referencia encontraron una solución Pareto óptima para asociarse.

En problemas en los que el frente de óptimo de Pareto completo es concentrado en una pequeña región o en aquellos problemas en los que existe una gran diferencia entre la cantidad de puntos de referencia iniciales y el tamaño de la población, el método para añadir nuevos

puntos de referencia es ineficiente debido a que no logra agregar la cantidad suficiente de puntos para distribuir uniformemente la población completa [Jain H. y Deb K., 2013].

Otro inconveniente en el algoritmo A-NSGA-III es que el método de Añadir Puntos de Referencia no permite generar puntos alrededor de los puntos de referencia en las esquinas del hiperplano. Existen más problemas originados por el método para añadir nuevos puntos de referencia, debido a estas limitaciones se implementó al A-NSGA-III un enfoque diferente en la actualización de los puntos de referencia, al A-NSGA-III con las modificaciones citadas se le nombró A^2-NSGA-III.

A diferencia de su próximo antecesor A-NSGA-III, la metaheurística A^2-NSGA-III actualiza su conjunto de puntos de referencia cada que se cumplen ciertas condiciones preestablecidas utilizando también un método simplex (M-1)-dimensional pero aplicado de distinta manera ya que en vez de generar M nuevos puntos de referencia y dejar el punto que se tomó como base como centro de ellos, genera los nuevos puntos eligiendo aleatoriamente una de las M formas distintas de crear los nuevos $M-1$ puntos, eliminando a la vez aquellos puntos de referencia que no tienen alguna solución asociada, repitiendo dicha adición y eliminación de puntos hasta que cada uno de ellos esté asociado solamente a una solución [Deb K. y Jain H., 2013].

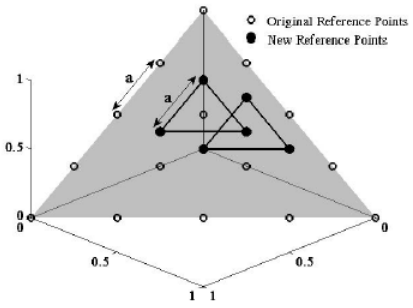


Fig. 4.5 Enfoque para añadir nuevos puntos de referencia en A-NSGA-III

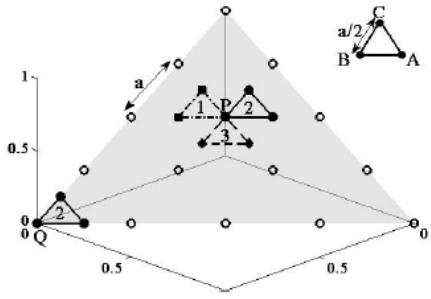


Fig. 4.6 Enfoque para añadir nuevos puntos de referencia en A^2-NSGA-III

4.3.2 Descripción de los métodos NSGA-III, A-NSGA-III y A^2-NSGA-III

Estas metaheurísticas se basan en combinar las soluciones con mejor calidad elegidas mediante la división de las soluciones en frentes al igual que en NSGA-II y soluciones que aportan diversidad, las cuales son seleccionadas a través de la asociación a un conjunto de puntos de referencia que son actualizados (en el caso de A-NSGA-III y A^2-NSGA-III) cuando durante cierto número de generaciones el máximo número de miembros de la población asociados a un punto de referencia no cambian. Constan básicamente de 9 métodos, cuatro de ellos (Generación de la población, Clasificación por no dominancia, Cruza y Mutación) trabajan de la misma forma que en NSGA-II como se muestra en la sección 4.2.1, mientras que el método Añadir Puntos de Referencia únicamente es utilizado en A-NSGA-III y A^2-NSGA-III con la diferencia que se menciona en la sección 4.3.1 [Jain H. y Deb K., 2013]:

1. **Selección.** Utiliza una estrategia de selección Torneo Binario, en la que se elige de la población de padres (de dos en dos) y de los cuales aquel que pertenece al mayor frente es agregado a la lista de padres que generarán a los hijos de esta generación, en caso de pertenecer al mismo frente se elige uno al azar.
2. **Método de normalización adaptativa de los miembros de la población**. Este método inicialmente identifica el punto ideal a través de las generaciones, utilizando para ello los valores de los objetivos por cartera, posteriormente traslada los objetivos por cartera de todos los miembros de la población que pertenezcan a los frentes de donde se seleccionarán los individuos que pasarán a la siguiente generación de padres (S_t), finalmente identifica los puntos extremos para calcular los interceptos que utiliza para terminar la normalización de los objetivos por cartera.
3. **Método de asociación**. Después de la normalización de cada objetivo de forma adaptativa basada en la extensión de los miembros de S_t en el espacio objetivo, se necesita asociar a cada miembro de la población con un punto de referencia. Para este propósito, se define una línea de referencia correspondiente a cada punto de referencia en el hiperplano uniendo el punto de referencia con el origen. A continuación, se calcula la distancia ortogonal de cada miembro de la población de S_t a cada una de las líneas de referencia. El punto de referencia cuya referencia es la línea más cercana a un miembro

de la población en el espacio objetivo normalizado se considera asociado con el miembro de la población.

4. **Operación de preservación-nicho.** Como ya se dijo anteriormente S_t es el conjunto de miembros de la población que después de generar los frentes de Pareto pasarán a la siguiente generación debido a su calidad, sin embargo S_t puede no pasar completo en caso de que su tamaño sea superior al número de miembros de la población, en tal caso se aplican los primeros dos métodos mencionados, posteriormente deben elegirse de los miembros de S_t que pertenecen al último frente que no pasará completo, aquellos miembros que por diversidad pasarán a la siguiente generación, esto se lleva a cabo con el método de preservación Nicho. Los pasos que realiza son:
    - **a.** Identificar todos los puntos de referencia que poseen el menor número de miembros de la población asociados a ellos (en caso de ser más de uno se elige uno de ellos de manera aleatoria $[j]$).
    - **b.** Identificar los miembros de la población asociados al punto j que pertenecen al último frente a añadir (F_l).
    - **c.** Si existe al menos un miembro asociado al punto j que pertenezca a F_l entonces:
        i. Si no existe al menos un miembro de S_t/F_l asociado a dicho punto de referencia, se añade a la siguiente población de padres aquel elemento de F_l asociado a este punto que cuente con la menor distancia al mismo.

        ii. En caso contrario, si existe algún miembro de S_t/F_l asociado a dicho punto de referencia se añade a la siguiente población de padres un elemento de F_l asociado a este punto de referencia elegido de forma aleatoria (en caso de ser más de uno).

        Independientemente de la manera en que se haya añadido el nuevo miembro a la siguiente población de padres, se incrementa el contador nicho correspondiente al punto de referencia j y se elimina del último frente el miembro ingresado.

d. En caso de que ningún miembro asociado al punto j se encuentre en F_l, entonces este punto de referencia ya no será tomado en cuenta para este proceso de selección para la población actual.

5. **Añadir puntos de referencia.** Este método es invocado después de que han pasado n generaciones en donde el máximo número de miembros de la población asociados a un punto de referencia no ha cambiado, entonces actualiza el conjunto de puntos de referencia empleando un método Simplex $(M-1)$ dimensional añadiendo $M-1$ nuevos puntos de referencia cerca del punto del que surgieron. La adición de estos puntos se hace solo cuando los puntos no existen y se encuentran además en el primer cuadrante, existen M formas de calcular los nuevos puntos de referencia, de ellas se elige una aleatoriamente, si los puntos calculados mediante la configuración elegida no son factibles o ya se encuentran dentro del conjunto de puntos, se prueba con una configuración distinta. Si se han agotado las configuraciones distintas y aun no existe una relación 1-1 entre los puntos de referencia y los miembros de la población, se repite el método reduciendo a la mitad la distancia (λ) entre los nuevos puntos a añadir y el punto generador. Por otra parte aquellos puntos de referencia que se encuentran sin miembros asociados son eliminados.

4.3.3 Algoritmos NSGA-III, A-NSGA-III y A²-NSGA-III

NSGA-III, A-NSGA-III y A²-NSGA-III

Entrada: Puntos de referencia estructurados Z^s o puntos de aspiración suministrados Z^a, población de padres P_t [Jain H. y Deb K., 2013].

Salida: P_{t+1}

1. $S_t = \emptyset, i = 1$
2. $Q_t = Recombinación + Mutación\ (P_t)$
3. $R_t = P_t \cup Q_t$
4. $(F_1, F_2, \dots) = Clasificación\_por\_no\_dominancia(R_t)$
5. **Repetir**
6. $S_t = S_t \cup F_i$ e $i = i + 1$
7. **Hasta que** $|S_t| \geq N$
8. Último frente a ser incluido: $F_l = F_i$
9. **Si** $|S_t| = N$ **entonces**
10. $P_{t+1} = S_t$, rompe el ciclo
11. **Si no**
12. $P_{t+1} = \bigcup_{j=1}^{l-1} F_j$
13. Puntos a ser elegidos de F_l: $K = N - |P_{t+1}|$
14. Normalizar objetivos y crear el conjunto de referencia Z^r: $Normalizar\ (f^n, S_t, Z^r, Z^s, Z^a)$
15. Asociar cada miembro de $s \in S_t$ con un punto de referencia: $[\pi(s), d(s)] = Asociar\ (S_t, Z^r)$ en donde:
    $\pi(s) =$ punto de referencia mas cercano y $d(s) =$ distancia entre s y $\pi(s)$
16. Computar contador nicho del punto de referencia $j \in Z^r$: $\rho_j = \sum_{s \in S_t/F_l} \big((\pi(s) = j)\ ?\ 1 : 0\big)$
17. Seleccionar K miembros uno a la vez de F_l para construir P_{t+1}: $Nicho(K, \rho_j, \pi, d, Z^r, F_l, P_{t+1})$
18. **Fin de Si** //Solo se implementa el paso 19 en
19. *Añadir_puntos_de_referencia*(Z^r, ρ, p, λ) // A-NSGA-III y A²-NSGA-III

Algoritmo 4.3 Algoritmo de las metaheurísticas NSGA-III, A-NSGA-III y A²-NSGA-III

4.3.4 Evaluación experimental de los algoritmos NSGA-III y A^2-NSGA-III

Debido a que en el experimento descrito en 4.2.4 ya se había efectuado una comparación entre los resultados arrojados por las metaheurísticas NSGA-III y A^2-NSGA-III comparándolas con los obtenidos de las metaheurísticas NSGA-II y MSS, en esta ocasión el experimento se enfocó en evaluar el desempeño de estos algoritmos implementándoles el sistema relacional de preferencias difusas propuesto por el Dr. Eduardo Fernández descrito en 2.5.1 al término de la última generación como un filtro final antes de presentar el resultado al DM, las condiciones experimentales fueron las mismas que en 4.2.4.

Los parámetros utilizados para el método de sobreclasificación difusa Electre III y en el sistema relacional de preferencias difusas fueron los siguientes:

Tabla 4.3 Parámetros empleados en Electre III y MPDF para la instancia de tres objetivos.

Para instancia de 3 objetivos			
Parámetros de Electre III		Parámetros del MPDF	
Peso por objetivo:	40-11-49	Simetría:	0.1
Umbrales de veto:	51,000-15,000-550	Asimetría:	0.4
Umbrales de indiferencia:	7,500-1,500-75	Umbral de credibilidad:	0.7

Tabla 4.4 Parámetros empleados en Electre III y MPDF para la instancia de nueve objetivos.

Para instancia de 9 objetivos			
Parámetros de Electre III		Parámetros del MPDF	
Peso por objetivo:	10-13-10-12-7-13-10-7-18	Simetría:	0.1
Umbrales de veto:	60,000-45,000-75,000-50,000-84,000-60,000-100,000-78,000-79,000	Asimetría:	0.4
Umbrales de indiferencia:	7,500-6,000-9,000-6,000 12,000-7,500-12,000-10,500-10,500	Umbral de credibilidad:	0.7

4.3.5 Resultados y conclusiones

Tabla 4.5 Resultados de la experimentación para la instancia de tres objetivos mediante el indicador de desempeño de Dominancia aplicando preferencias.

	Corrida	NSGA-II	NSGA-III	A^2-NSGA-III	MSS
% de soluciones no superadas	1	66% (2 de 3)	**100% (6 de 6)**	**100% (2 de 2)**	**100% (1 de 1)**
	2	100% (3 de 3)	100% (2 de 2)	100% (2 de 2)	100% (3 de 3)
	3	100% (2 de 2)	100% (1 de 1)	100% (1 de 1)	100% (1 de 1)
	4	100% (7 de 7)	100% (3 de 3)	100% (1 de 1)	100% (3 de 3)
	5	100% (1 de 1)	100% (1 de 1)	100% (3 de 3)	100% (2 de 2)
	6	66% (2 de 3)	**100% (4 de 4)**	**100% (5 de 5)**	**100% (2 de 2)**
	7	100% (4 de 4)	100% (2 de 2)	100% (8 de 8)	100% (3 de 3)
	8	100% (6 de 6)	100% (1 de 1)	100% (3 de 3)	100% (1 de 1)
	9	100% (5 de 5)	100% (2 de 2)	100% (2 de 2)	100% (1 de 1)
	10	**100% (3 de 3)**	33% (1 de 3)	**100% (9 de 9)**	**100% (2 de 2)**
	11	100% (7 de 7)	100% (4 de 4)	100% (2 de 2)	100% (1 de 1)
	12	100% (3 de 3)	100% (2 de 2)	100% (2 de 2)	100% (2 de 2)
	13	**100% (6 de 6)**	50% (2 de 4)	**100% (2 de 2)**	**100% (1 de 1)**
	14	100% (3 de 3)	100% (2 de 2)	100% (5 de 5)	100% (1 de 1)
	15	100% (1 de 1)	100% (4 de 4)	100% (3 de 3)	100% (1 de 1)
	16	**100% (5 de 5)**	66% (2 de 3)	**100% (3 de 3)**	**100% (1 de 1)**
	17	**100% (5 de 5)**	50% (1 de 2)	**100% (1 de 1)**	**100% (1 de 1)**
	18	**100% (2 de 2)**	75% (3 de 4)	**100% (2 de 2)**	**100% (2 de 2)**
	19	100% (8 de 8)	100% (3 de 3)	100% (2 de 2)	100% (1 de 1)
	20	100% (4 de 4)	100% (2 de 2)	100% (8 de 8)	100% (1 de 1)
	21	100% (3 de 3)	100% (2 de 2)	100% (3 de 3)	100% (1 de 1)
	22	100% (6 de 6)	100% (2 de 2)	100% (5 de 5)	100% (1 de 1)
	23	**100% (5 de 5)**	50% (1 de 2)	**100% (2 de 2)**	**100% (3 de 3)**
	24	**100% (4 de 4)**	**100% (5 de 5)**	75% (3 de 4)	**100% (5 de 5)**
	25	100% (3 de 3)	100% (1 de 1)	100% (5 de 5)	100% (2 de 2)
	26	100% (4 de 4)	100% (1 de 1)	100% (2 de 2)	100% (6 de 6)
	27	100% (7 de 7)	50% (1 de 2)	100% (4 de 4)	100% (2 de 2)
	28	**100% (2 de 2)**	75% (3 de 4)	**100% (5 de 5)**	**100% (1 de 1)**
	29	100% (5 de 5)	100% (4 de 4)	100% (4 de 4)	100% (2 de 2)
	30	100% (2 de 2)	100% (1 de 1)	100% (4 de 4)	100% (1 de 1)

Tabla 4.6 Resultados de la experimentación para la instancia de nueve objetivos mediante el indicador de desempeño de Dominancia aplicando preferencias.

	Corrida	NSGA-II	NSGA-III	A^2-NSGA 3	MSS
% de soluciones no superadas	1	62% (5 de 8)	**100% (1 de 1)**	**100% (1 de 1)**	**100% (2 de 2)**
	2	25% (1 de 4)	**100% (1 de 1)**	0% (0 de 1)	**100% (3 de 3)**
	3	83% (5 de 6)	**100% (1 de 1)**	**100% (1 de 1)**	**100% (3 de 3)**
	4	0% (0 de 1)	57% (8 de 14)	**100% (7 de 7)**	**100% (1 de 1)**
	5	**100% (4 de 4)**	71% (5 de 7)	**100% (1 de 1)**	**100% (1 de 1)**
	6	100% (6 de 6)	100% (6 de 6)	100% (1 de 1)	100% (1 de 1)
	7	100% (1 de 1)	69% (9 de 13)	28% (2 de 7)	100% (1 de 1)
	8	0% (0 de 1)	66% (2 de 3)	**100% (1 de 1)**	**100% (3 de 3)**
	9	**100% (8 de 8)**	33% (2 de 6)	75% (6 de 8)	**100% (3 de 3)**
	10	100% (2 de 2)	100% (1 de 1)	100% (1 de 1)	100% (1 de 1)
	11	**100% (1 de 1)**	75% (6 de 8)	**100% (1 de 1)**	**100% (1 de 1)**
	12	0% (0 de 1)	**100% (1 de 1)**	0% (0 de 1)	**100% (3 de 3)**
	13	**100% (2 de 2)**	75% (6 de 8)	**100% (1 de 1)**	**100% (1 de 1)**
	14	**100% (4 de 4)**	0% (0 de 4)	80% (4 de 5)	**100% (1 de 1)**
	15	100% (1 de 1)	100% (1 de 1)	100% (1 de 1)	100% (1 de 1)
	16	80% (4 de 5)	75% (6 de 8)	54% (6 de 11)	**100% (1 de 1)**
	17	**100% (1 de 1)**	**100% (1 de 1)**	71% (5 de 7)	**100% (3 de 3)**
	18	**100% (4 de 4)**	44% (4 de 9)	**100% (5 de 5)**	**100% (1 de 1)**
	19	**100% (3 de 3)**	0% (0 de 7)	0% (0 de 1)	**100% (1 de 1)**
	20	100% (1 de 1)	100% (1 de 1)	100% (2 de 2)	100% (1 de 1)
	21	**100% (2 de 2)**	**100% (1 de 1)**	83% (5 de 6)	**100% (1 de 1)**
	22	**100% (1 de 1)**	80% (4 de 5)	87% (7 de 8)	**100% (3 de 3)**
	23	80% (4 de 5)	0% (0 de 1)	**100% (1 de 1)**	**100% (1 de 1)**
	24	**100% (7 de 7)**	72% (8 de 11)	**100% (1 de 1)**	**100% (1 de 1)**
	25	**100% (1 de 1)**	25% (2 de 8)	50% (3 de 6)	**100% (3 de 3)**
	26	**100% (1 de 1)**	60% (6 de 10)	**100% (1 de 1)**	**100% (2 de 2)**
	27	75% (3 de 4)	**100% (1 de 1)**	**100% (3 de 3)**	**100% (1 de 1)**
	28	**100% (4 de 4)**	0% (0 de 1)	0% (0 de 1)	**100% (2 de 2)**
	29	**100% (1 de 1)**	81% (9 de 11)	**100% (5 de 5)**	**100% (1 de 1)**
	30	**100% (1 de 1)**	66% (10 de 15)	**100% (1 de 1)**	**100% (1 de 1)**

En las tablas 4.5 y 4.6 puede observarse el resultado de la experimentación para las instancias de tres y nueve objetivos respectivamente, las celdas en color amarillo corresponden a las metaheurísticas con mejor desempeño en la corrida correspondiente cuando más de una tuvo el mismo porcentaje en el indicador de calidad de dominancia con el que se evaluaron, por otro lado las celdas en color verde indican la metaheurística que en esa corrida tuvo el mejor desempeño superando a todas las demás.

Como se observa el algoritmo NSGA-III no ofrece muy buenos resultados en comparación con los otros tres algoritmos en este experimento quedando atrás incluso de su predecesor NSGA-II, sin embargo es notorio que con la instancia para tres objetivos NSGA-II, A^2-NSGA-III y MSS tienen un desempeño equilibrado y muy similar, aunque en la tabla 4.6 se puede observar como al incrementarse la complejidad del problema con una instancia de nueve objetivos el desempeño de NSGA-II y A^2-NSGA-III decrece y queda a la par, mientras que los resultados brindados por MSS mejoran.

Se puede concluir que NSGA-II presenta mejores resultados que su sucesor NSGA-III, sin embargo su desempeño es similar al de la metaheurística A^2-NSGA-III e inferior al brindado por la metaheurística MSS. Debido a que NSGA-II fue diseñado para la solución de problemas multiobjetivo (hasta tres objetivos) y A^2-NSGA-III tiene características que le brindan un mejor desempeño en problemas de muchos objetivos (más de tres) probablemente a mayor número de objetivos en un problema NSGA-II presente resultados de menor calidad frente a los de A^2-NSGA-III, mientras que posiblemente A^2-NSGA-III comience a ofrecer un desempeño similar al de MSS.

Por otro lado, observando las tablas 4.1 y 4.2 del experimento en 4.2.4 y comparándolas con los resultados obtenidos en las tablas 4.3 y 4.4 se observa como los resultados arrojados por la metaheurística NSGA-II se ven más similares a los de A^2-NSGA-III, esto se debe a la implementación que se realizó del sistema relacional de preferencias difusas, del cual se concluye que al aplicarse sobre un conjunto de soluciones, toma aquellas que se encuentran más apegadas a los criterios del DM y desecha aquellas de menor calidad haciendo parecer que el desempeño de las metaheurísticas es más similar de lo que sería si se presentara el conjunto de soluciones finales completo como paso en el experimento 4.2.4.

4.4 Introducción a la Búsqueda Dispersa Multiobjetivo

Estado del arte de la Búsqueda Dispersa Multiobjetivo

En los últimos años, la solución de problemas que requieren optimizar más de una función ha recibido una gran atención, estando este interés motivado en gran medida por la naturaleza multiobjetivo de los problemas del mundo real [Nebro A. et. al, 2007].

Las técnicas deterministas no son aplicables, siendo necesario, por tanto, usar métodos estocásticos. Entre ellos, las metaheurísticas aparecen como una familia de técnicas aproximadas que son muy utilizadas en muchos campos para resolver problemas de optimización; en particular, el uso de algoritmos evolutivos para resolver Problemas Multiobjetivo (MOPs, por su sigla en inglés) ha experimentado un crecimiento significativo en los últimos años, dando lugar a la aparición de una gran variedad de algoritmos metaheurísticos, como NSGA-II, NSGA-III, A-NSGA-III, A^2-NSGA-III, SPEA2, PAES, etc.

SS es una metaheurística que puede ser considerada como un Algoritmo Evolutivo en el sentido de que incorpora el concepto de un conjunto de soluciones que evoluciona. SS evita generalmente el uso de componentes aleatorios, no obstante los operadores típicos de los Algoritmos Evolutivos como los operadores de cruza y mutación se pueden adaptar a la filosofía de esta técnica.

Si bien SS ha sido utilizada con éxito para resolver una gran variedad de problemas de optimización mono-objetivo, su aplicación a problemas multiobjetivo es marginal. Se tiene registro del uso de esta metaheurística para dar solución a problemas multiobjetivo tales como los pertenecientes a la familia ZDT. En el trabajo desarrollado por Nebro [Nebro A. et. al, 2007] se añadieron operadores que provienen del ámbito de los algoritmos genéticos que se utilizan para cruza y mutación. El método SBX se aplica para implementar el método de combinación de soluciones, por otro lado, como método de mejora utiliza la estrategia de mutación de (1+1) EA. Este algoritmo híbrido recibe el nombre de Archive-based hYbrid Scatter Search (AbYSS), y se evalúa comparando su desempeño con el de los algoritmos NSGA-II y SPEA2, los resultados reportados muestran la superioridad del algoritmo híbrido [Nebro A. et. al, 2007].

En resumen, la metaheurística SS se adapta en un algoritmo enfocado a la solución de un problema multiobjetivo mediante la incorporación de algunos cambios importantes tales como: modificar las estructuras de almacenamiento para incluir múltiples objetivos, incorporar un criterio multiobjetivo para evaluar la calidad de las soluciones tal como "el criterio de dominación", también es posible incluir o combinar algunas estrategias de algoritmos multiobjetivo para medir la diversidad, tales como el estimador de densidad de SPEA2 [Zitzler E. et. al, 2001], o el grid adaptativo de PAES conservando el mismo algoritmo base de SS. A esta adaptación se le conoce como Búsqueda Dispersa Multiobjetivo.

Descripción del método SS

SS se basa en combinar las soluciones que aparecen en el llamado conjunto de referencia. Este conjunto almacena las buenas soluciones que se han ido encontrando durante el proceso de búsqueda. Cabe destacar que una solución se considera buena no solo en el sentido de la calidad de ésta, sino que también se considera la diversidad que ésta aporta al conjunto de referencia. SS consta básicamente de cinco elementos o métodos [Martí R. y Laguna M., 2003]:

1. **Método de generación de P.** Un generador de soluciones diversas. El método se basa en generar un conjunto P de soluciones diversas (alrededor de 100), del que se extrae un subconjunto pequeño (aproximadamente el 10%) al que se denomina conjunto de referencia $RefSet$.
2. **Método de mejora.** Típicamente se trata de un método de búsqueda local para mejorar las soluciones, tanto del conjunto de referencia como las combinadas antes de estudiar su inclusión en el conjunto de referencia. Es importante destacar que en las implementaciones donde se manejen soluciones no factibles, éste método debe ser capaz de, a partir de una solución no factible, obtener una que sea factible y después intentar mejorar su valor. Si el método no logra mejorar a la solución inicial, se considera que el resultado es la propia solución inicial.
3. **Método de actualización del conjunto referencia.** Es un procedimiento para crear y actualizar el conjunto de referencia $RefSet$. A partir del conjunto de soluciones diversas P se extrae el conjunto de referencia según el criterio de contener soluciones de calidad

y diferentes entre sí (calidad y diversidad). Las soluciones en este conjunto están ordenadas de mejor a peor respecto de su calidad.

   a. **Creación.** Iniciamos el conjunto de referencia con las $\frac{b}{2}$ ($|RefSet| = b$) mejores soluciones de P. Las $\frac{b}{2}$ restantes se extraen de P con el criterio de maximizar la mínima distancia con las ya incluidas en el conjunto de referencia. Para ello debemos de definir previamente una función de distancia en el problema.

   b. **Actualización.** Las soluciones fruto de las combinaciones pueden entrar en el conjunto de referencia y reemplazar a alguna de las ya incluidas, en el caso de que las mejoren. Así pues, el conjunto de referencia mantiene un tamaño b constante, pero el valor de sus soluciones va mejorando a lo largo de la búsqueda. En implementaciones sencillas, la actualización de este conjunto se realiza únicamente por calidad, aunque se puede hacer también por diversidad.

4. **Método de generación de subconjuntos.** Genera subconjuntos de $RefSet$ a los que se aplicará el método de combinación. SS se basa en examinar de una forma bastante exhaustiva todas las combinaciones de $RefSet$. Éste método especifica la forma en que se seleccionan los subconjuntos para aplicarles el método de combinación. Una implementación sencilla, utilizada a menudo, consiste en restringir la búsqueda a parejas de soluciones. Así el método considera todas las parejas que se pueden formar con los elementos de $RefSet$ y a todas ellas le aplica el método de combinación.
5. **Método de combinación de soluciones.** SS se basa en combinar todas las soluciones del conjunto de referencia. Para ello, se consideran los subconjuntos formados por el método del paso 4, y se le aplica el método de combinación. La solución o soluciones que se obtienen de esta combinación pueden ser inmediatamente introducidas en el conjunto de referencia o almacenadas temporalmente en una lista hasta terminar de realizar todas las combinaciones y después ver qué soluciones entran en éste.

Estrategias de diversidad de SS

Las operaciones principales de este algoritmo trabajan sobre el conjunto de referencia o $RefSet$, cuyo tamaño se recomienda en [Martí R. y Laguna M., 2003] que sea de aproximadamente 10% del tamaño total de P. El llenado del conjunto referencia se realiza en dos etapas, la primera añade a $RefSet$ aquellas soluciones cuyos valores objetivo es más alto; la segunda agrega a $RefSet$ aquellas soluciones en $P \backslash RefSet$ que se encuentren más alejadas del conjunto de referencia (considerando la distancia Euclidiana a la solución más cercana del conjunto de referencia), este último grupo de soluciones proporcionarán a la búsqueda en la siguiente iteración. Finalmente la aplicación de los métodos de generación de subconjuntos, combinación y mejora se aplican sobre las soluciones del conjunto de referencia sin considerar si forman parte de las soluciones que ingresaron al $RefSet$ por su calidad o para diversificar.

Algoritmo SS

SS

1. Comenzar con $P = 0$. Utilizar el *método_de_generación* para construir una solución y el método de mejora para tratar de mejorarla; sea x la solución obtenida. Si $x \notin P$ entonces añadir x a P, en otro caso, rechazar x. Repetir esta etapa hasta que P tenga un tamaño prefijado [Martí R. y Laguna M., 2003].
2. Construir el **conjunto de referencia** $RefSet = \{x^1, \dots, x^b\}$ con las $\frac{b}{2}$ mejores soluciones de P y las $\frac{b}{2}$ soluciones de P más diversas a las ya incluidas.
3. Evaluar las soluciones en $RefSet$ y ordenarlas de mejor a peor respecto a la función objetivo.
4. $NuevaSolución = \text{TRUE}$
5. **Mientras** (Queden subconjuntos sin examinar)
6. $NuevaSolución = \text{FALSE}$
7. Generar los subconjuntos de $RefSet$ en los que haya al menos una nueva solución.
8. **Mientras** (Queden subconjuntos sin examinar)
9. Seleccionar un subconjunto y etiquetarlo como examinado.
10. Aplicar el *método_de_combinación* a las soluciones del subconjunto.

11. Aplicar el *método_de_mejora* a cada solución obtenida por combinación. Sea x la solución mejorada:
12. **Si** ($f\ (x) < f\ (x^b)$ y x no está en $RefSet$) **entonces**
13. $x^b = x$ y reordenar $RefSet$.
14. $NuevaSolución =$ TRUE.
15. **Fin de Si**

Algoritmo 4.4 Metaheurística SS.

4.4.1 Descripción del método Búsqueda Dispersa Multiobjetivo

MSS a diferencia de SS basa su estrategia en combinar las soluciones que aparecen en un par de conjuntos de referencias a los que llama $RefSet_1$ y $RefSet_2$. Estos conjuntos almacenan las buenas soluciones que se han ido encontrando durante el proceso de búsqueda, el primero ($RefSet_1$) almacena las soluciones basándose en su calidad y el segundo conjunto ($RefSet_2$) almacena las soluciones basándose en su diversidad empleando el estimador de densidad utilizado por SPEA2. MSS consta de ocho métodos principales, cinco de ellos trabajan de la misma forma que en SS (Generación de P, Combinación, Mejora, Generación de subconjuntos, Combinación de soluciones) con la única diferencia de que en lugar de trabajar con un conjunto de referencia, utiliza dos conjuntos a los cuales les aplica los mismos pasos pero tratándolos de manera independiente. El método de actualización de los conjuntos referencia tiene cambios significativos que se describen en la siguiente sección al igual que dos métodos que no utiliza SS.

4.4.2 Estrategias de diversidad de la Búsqueda Dispersa Multiobjetivo

A diferencia de la estrategia utilizada en la SS, MSS trabaja de manera independiente las soluciones de calidad y las soluciones que proporcionan diversidad, a continuación se describen los métodos en los que se detalla la manera en que MSS evita el estancamiento en óptimos locales y que además no se encuentran (en el caso del método Gestión del archivo externo) o trabajan de diferente manera en SS.

1. **Método de actualización de los conjuntos referencia.** Genera a partir del conjunto P los subconjuntos $RefSet_1$ y $RefSet_2$, después de que P es creado o reiniciado, también se encarga de actualizar los dos conjuntos de referencia. El conjunto $RefSet_1$ contiene las mejores soluciones aplicando un criterio de calidad, por lo que guía la búsqueda hacia el frente de Pareto, mientras que el conjunto $RefSet_2$ contiene las soluciones encargadas de evitar el estancamiento promoviendo la diversificación de las soluciones en el frente.

   Estos conjuntos se obtienen inicialmente del conjunto P. Los criterios para la construcción de $RefSet_1$ y $RefSet_2$ se describen en seguida:

   Criterios de calidad para $RefSet_1$

   - Se utiliza el estimador de densidad de SPEA2 [Zitzler E. et. al, 2001] para ordenar las soluciones del conjunto P, tomando las primeras b_1 soluciones de P para formar $RefSet_1$. En la actualización la peor solución en $RefSet_1$ será sustituida por las soluciones obtenidas en el proceso de Mejora, siempre y cuando tengan una mejor calidad de acuerdo al estimador de densidad de SPEA2.

   Criterios de diversidad para $RefSet_2$

   - Se utiliza la distancia Euclidiana para seleccionar las soluciones de $P \backslash RefSet_1$ que formarán a $RefSet_2$. Para cada solución de $P \backslash RefSet_1$ se calcula la distancia a cada una de las soluciones en $RefSet_1$, la mínima de estas distancias es el indicador de diversidad (ID) asociado a esa solución, finalmente se toman las b_2 soluciones que tienen los mayores valores para este indicador de diversidad.

   En el caso de la actualización, las soluciones generadas en el método de combinación pasan al método de mejora, y una vez mejoradas se incluyen en alguno de los conjuntos de referencia aplicando el criterio de calidad o diversidad correspondiente:

   $RefSet_1$:

   - Si tiene una mejor calidad, que la peor solución en $RefSet_1$, la sustituye.

- Si tiene la misma calidad que la peor solución en $RefSet_1$, la nueva solución es insertada y la peor solución es enviada al archivo externo.

$RefSet_2$:

- Si su ID es mayor que el menor ID de todas las soluciones en $RefSet_2$, la nueva solución sustituirá a aquella que tenga el menor ID.
- Si su ID es igual al menor ID de todas las soluciones en $RefSet_2$, la nueva solución es insertada en $RefSet_2$ y la solución con menor ID es enviada al archivo externo.

2. **Gestión del archivo externo.** El principal objetivo del archivo externo es mantener un registro histórico de las soluciones no dominadas encontradas durante la búsqueda, a la vez que se mantienen aquellas que brindan una mejor distribución sobre el frente de Pareto [Nebro A. et. al, 2008].

ARCHIVO EXTERNO

Entrada: Solución no dominada S del ciclo principal de MSS.

Salida: W //conjunto de soluciones no dominadas

1. $bandera = falso$ //variable utilizada para saber si alguna solución del archivo externo dominó a S
2. **Para** cada solución en el archivo externo $w \in W$ **hacer**
3. **Si** w domina a S **entonces**
4. $bandera = verdadero$
5. **Fin de Si**
6. **Si** $bandera = falso$ **entonces**
7. **Para** cada solución en el archivo externo $w \in W$ **hacer**
8. **Si** S domina a w **entonces**
9. $Eliminar(w)$
10. **Fin de Si**
11. $Agregar(\,)$ //S a W
12. **Fin de Si**
13. **Si** W está lleno **entonces**
14. *DistanciaDeCrowding( )* //para determinar qué solución será suprimida
15. **Fin Si**

Algoritmo 4.5 Añadir solución al archivo externo.

3. **Reinicialización de P.** Su función es evitar una convergencia prematura dando oportunidad al algoritmo de explorar en el espacio de soluciones. Este método se ejecuta si durante la ejecución del proceso de Actualización del Conjunto Referencia no se encuentra al menos una nueva solución que cumpla los criterios de calidad o diversidad para incorporarse en el conjunto referencia y al mismo tiempo no se han llevado a cabo el número de evaluaciones de la función objetivo requeridas para la finalización del algoritmo, en tal caso este método realiza lo siguiente: Elimina todas las soluciones contenidas en P, añade aquellas soluciones contenidas en $RefSet_1$, se extraen las n (el mínimo entre $\frac{|P|}{2}$ y el tamaño del archivo externo) mejores soluciones del archivo externo de acuerdo a su Distancia de Crowding y se incorporan a P, finalmente se construye una

solución de referencia (S_r) con los mejores de cada objetivo encontrados en las soluciones que se han añadido en P a partir de esa solución se generan soluciones aleatorias que serán incluidas en P únicamente si superan a S_r en al menos un objetivo (esto se lleva a cabo hasta completar el tamaño de P).

4.5 Búsqueda Dispersa Multiobjetivo Guiada por Preferencias

A través de una investigación y análisis semejante a lo mostrado en las secciones 4.2.4 y 4.3.4, se desarrolló una mejora al algoritmo de MSS cuyos resultados fueron superiores a lo que se esperaba y que deja un gran precedente para futuras investigaciones sea de esta modificación a la metaheurística o alguna adaptación similar que se realice a otro algoritmo.

En las tablas 4.1 y 4.2 se vio que la metaheurística MSS presentaba resultados ligeramente mejores a los de NSGA-II, NSGA-III y A^2-NSGA-III tanto para la instancia de tres objetivos como para la de nueve objetivos, posteriormente en las tablas 4.5 y 4.6 se resaltó el hecho de que al implementar el sistema relacional de preferencias difusas se lograba obtener lo mejor de un conjunto de soluciones al seleccionar de ellas a las más cercanas a lo que el DM desea, uniendo estos dos puntos se llevó a cabo una mejora a MSS para que superara la calidad de sus propias soluciones y de esta manera destacara por un margen mayor sobre las soluciones del resto de las metaheurísticas mencionadas, a dicho algoritmo mejorado se le llamó Búsqueda Dispersa Multiobjetivo Guiada por Preferencias (MSS-GP, por su sigla en inglés).

Los cambios mencionados se efectuaron en dos de los métodos principales de la metaheurística base en los que se determinó que tendría un mayor impacto sobre el resultado final, los métodos Generación de P y Reinicilización de P. A continuación se describe detalladamente las modificaciones efectuadas sobre los algoritmos de ambos métodos.

4.5.1 Estrategias de diversidad de MSS-GP

En esta metaheurística se utiliza la misma estrategia de diversidad planteada para MSS descrita en la sección 4.4. La única diferencia que impacta sobre todo en la calidad de las soluciones es la utilización de una solución referencia (S_r) calculada apoyándose en el uso del sistema relacional de preferencias difusas descrito en la sección 2.5.1, el cual se implementa en dos diferentes métodos del algoritmo: en la Generación del conjunto P, en donde en vez de generar N soluciones aleatorias factibles, se generan $N * 5$ soluciones de ese tipo y se les aplica el sistema relacional de preferencias difusas para obtener de dicho conjunto un subconjunto de soluciones sobre las que no haya preferencia estricta de alguna otra solución hacia ellas, de dicho subconjunto se toman los datos extremos por objetivo (los mayores en caso de maximización, los menores en el caso de minimización) formando la S_r. Primero se vacía P, posteriormente se vacían las soluciones que forman parte del subconjunto de soluciones resultante del sistema relacional de preferencias, finalmente se completa el conjunto inicial de soluciones P generando soluciones aleatorias factibles e incluyendo solamente aquellas que sean superen en al menos un objetivo a la S_r. El otro método que varía con respecto a MSS es la Reinicialización de P, en el que se realiza algo similar a lo ocurrido en el método de Generación del conjunto P, a diferencia de que los datos extremos por objetivo que conforman la S_r en este caso se toman los peores casos extremos, es decir, los menores valores por objetivo en el caso de maximización o los mayores valores por objetivo en el caso de minimización.

4.5.2 Algoritmo MSS-GP

Este algoritmo general es el mismo de MSS descrito en 4.4 con las respectivas modificaciones puntualizadas en 4.4.1, a continuación se mostrarán los algoritmos de los métodos modificados que transforman MSS en MSS-GP y se describirán los cambios.

MÉTODO GENERAR P

Entrada: Tamaño de N (número de soluciones iniciales para P)

Salida: P

1. cont = 0
2. **Mientras** $(cont < (N * 5))$ // Originalmente era $(cont < N)$
3. Genera aleatoriamente una *nueva_solución* factible
4. **Si** (*nueva_solución* no es nulo)
5. **Si** (P no contiene a *nueva_solución*)
6. Se agrega nuevo *nueva_solución* a P
7. Se incrementa *cont* en una unidad
8. **Fin de Si**
9. **Fin de Si**
10. Calcular los objetivos de cada solución en P
11. $indices = MPDF(), cont = 0$
12. **Mientras** $(cont < indices.size())$
13. $P\_aux = P[índice[cont++]];$
14. $P = población\_aux$
15. $solución\_referencia = Calcula\_solución\_referencia()$
16. $cont = 0$
17. **Mientras** $(cont < N)$
18. $solución = genera\_solución\_aleatoria()$
19. **Si** (*solución* no es nula)
20. **Si** (*solución* supera en al menos un objetivo al *punto_de_referencia*)
21. $P[cont++] = solución$
22. **Fin de Si**
23. **Fin de Si**

Algoritmo 4.6 Método para la generación de P.

En este algoritmo los pasos 1 al 8 utilizan los mismos procedimientos que en el algoritmo original de este método, a diferencia de una pequeña modificación en el paso 2, en la que en vez de generar N soluciones aleatorias como hace MSS se genera una P cinco veces más grande con el fin de dar una mayor posibilidad a que el tamaño del subconjunto de soluciones en P devuelto por el sistema relacional de preferencias difusas sea un poco mayor, ya que como se vio en las tablas 4.5 y 4.6 este método devuelve a partir de un conjunto de tamaño entre 100 y 165 elementos, un subconjunto filtrado de tamaño uno a siete y en raras ocasiones supera los 10 elementos.

En el paso 9 la variable índices obtiene el índice de cada elemento en P perteneciente al subconjunto que más se apega a los criterios del DM de acuerdo al método MPDF( ) el cual implementa el sistema relacional de preferencias difusas (en este caso el propuesto por el Dr. Eduardo Fernández descrito en 2.5.1). Los pasos 10 y 11 describen un ciclo en el que se genera el subconjunto de soluciones en P que pertenecen a los elementos en índices, posteriormente en el paso 12 se hace una sustitución en la que se suple P por P_aux. En el paso 13 se asigna a la S_r el punto calculado mediante el método *calcula_solución_referencia*() que devuelve los valores máximos de cada objetivo de todas las soluciones en P.

En los pasos 15 al 19 se generan soluciones aleatorias que en caso de ser factibles y superar en al menos un objetivo a la S_r se incluirán en P hasta completarla.

MÉTODO REINICIALIZACIÓN DE P

Entrada: $RefSet_1$, P, $archivo\_externo$, N (número de soluciones totales en P)

Salida: P

1. Eliminar todas las soluciones de P
2. Añadir todas las soluciones de $RefSet_1$ a P
3. Se calcula k que es el número de soluciones que pasarán del *archivo_externo* a P, siendo el mínimo entre $|\frac{P}{2}|$ y $|archivo\_externo|$
4. **Si** (k es mayor o igual que $N - |P|$)
5. Aplicar *distancia_de_crowding* a las soluciones del *archivo_externo*
6. **Mientras**($|P|$ sea menor que N)
7. Agregar una solución del *archivo_externo*
8. **Si no**
9. Agregar k soluciones del *archivo_externo* a P
10. Generar una *solución_referencia* con los peores casos por objetivo tomando en cuenta las soluciones que se encuentran en P
11. **Mientras** ($|P|$ sea menor que N)
12. Generar aleatoriamente una *nueva_solución*
13. **Si**(*nueva_solución* supera en al menos un obj. a la *solución_referencia*)
14. Se añade *nueva_solución* a P

Algoritmo 4.7 Reinicialización de P.

Los pasos uno al nueve son los mismos que en el algoritmo original, del paso 10 en adelante se encuentra la modificación propuesta. En el paso 10 se calcula una S_r que es formada con los peores casos por objetivo de todas las soluciones que se almacenan en P, se decidió escoger los peores casos debido a que después de que el algoritmo ha iterado cierto número de veces, las soluciones son tan buenas que superar una S_r con los mejores casos requiere de un costo computacional muy grande y consume una gran cantidad de tiempo; mediante la experimentación se observó que al tomar los peores casos, con el paso de las iteraciones las soluciones generadas en base a la S_r seguían siendo muy buenas en comparación con el resto

de soluciones en P, de esta forma se incluyen para brindar diversidad a las soluciones ya encontradas en P.

Después de obtener la S_r, los pasos 11 al 14 definen un ciclo que se ejecutará hasta que el tamaño de P sea igual a la N definida en un principio como número de soluciones en nuestra estructura principal. En este ciclo se generan soluciones de manera aleatoria las cuales se irán incluyendo en P únicamente si son factibles, no se encuentran ya en dicha estructura y además superan en al menos un objetivo a la S_r calculada en el paso 10.

4.5.3 Evaluación experimental de los algoritmos MSS y MSS-GP

Cuatro evaluaciones fueron llevadas a cabo la primera de ellas con el fin de comprobar la correctez y desempeño de MSS y de MSS-GP. Las tres evaluaciones se enfocaron en probar el desempeño de MSS-GP comparando sus resultados con los de A^2-NSGA-III, MSS y NO-ACO. Las condiciones de hardware y software en los cuatro experimentos fueron las mismas que las del experimento descrito en 4.2.4. Los valores de los parámetros utilizados por A^2-NSGA-III en la segunda evaluación son los descritos para ésta metaheurística en el experimento descrito en 4.2.4. NO-ACO utilizó la configuración descrita en [Cruz L. et. al, 2013]. Se realizó un ajuste de parámetros para las metaheurísticas MSS y MSS-GP obteniéndose los siguientes valores:

Un tamaño de 100 para el conjunto P, 30 para $RefSet_1$, 50 para $RefSet_2$ y 50 para el archivo externo. Una probabilidad de combinación de 100% mediante el método de Combinación Uniforme. En el caso de MSS se ejecutan intercaladamente los método de combinación Uniforme y SBX (en donde SBX se ejecuta cada cinco iteraciones). Para el método de mejora se aplica la estrategia (1+1) EA con un porcentaje de probabilidad de 1%. Con el fin de comparación con los algoritmos del estado del arte se utilizan 700 mil evaluaciones de la función objetivo como condición de terminación [Nebro A. et. al, 2008]. Para cada experimento se realizaron 30 ejecuciones con cada una de las metaheurística evaluadas.

4.5.3.1 MSS y A^2-NSGA-III utilizando las instancias estándar de la familia DTLZ

En este experimento se realiza una evaluación comparativa del desempeño de MSS básico y la metaheurística A^2-NSGA-III [Jain H. y Deb K., 2013] así como la correctez de sus resultados. En el MSS básico los métodos Generación de *P* y Reinicialización de *P* no incluyen el MPDF ni el manejo de la S_r; en la Generación de *P* se obtienen las soluciones aleatoriamente verificando únicamente que sean factibles y no repetidas, mientras que *P* se reinicializa con las soluciones en $RefSet_1$, las soluciones contenidas en el archivo externo y en caso de que *P* no esté aun completo incorpora soluciones generadas aleatoriamente hasta completarlo.

Para este experimentos se utilizan los problemas estándar DTLZ1, DTLZ2, DTLZ3 y DTLZ7 pertenecientes a la familia DTLZ, con tres objetivos en todos los casos. A continuación se describen los problemas evaluados.

DTLZ1

Es un problema de prueba simple que tiene *M* objetivos y cuenta con un frente óptimo de Pareto verdadero. Este es un problema de minimización que se define mediante las siguientes funciones [Coello C. et. al, 2007]:

Minimizar:

$$f_1(x) = \frac{1}{2}x_1x_2 \dots x_{M-1}(1 + g(x_M)),$$

$$f_2(x) = \frac{1}{2}x_1x_2 \dots (1 - x_{M-1})(1 + g(x_M)),$$

$\dots$

$$f_{M-1}(x) = \frac{1}{2}x_1(1 - x_2)(1 + g(x_M)),$$

$$f_M(x) = \frac{1}{2}(1 - x_1)\big(1 + g(x_M)\big),$$

Sujeto a: $0 \le x_i \le 1, \forall \mathrm{i} = 1, 2, 3, \dots, n$

En donde: $g(x_M) = 100\big[|x_M| + \sum_{x_i \in X_M}(x_i - 0.5)^2 - cos\big(20\pi(x_i - 0.5)\big)\big]$

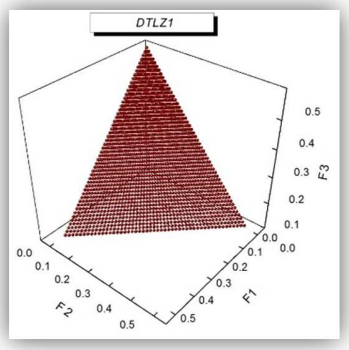


Fig. 4.7 DTLZ1

Las soluciones Pareto óptimas corresponden a $x_M^* = 0$ y a los valores de la función objetivo sobre el hiperplano lineal $\sum_{m=1}^{M} = 0.5$. El número de variables está dado por $n = M + k - 1$, en donde $k = 5$ como se sugiere en [Coello C. et. al, 2007].

DTLZ2

Este problema de prueba [Coello C. et. al, 2007] consiste en minimizar:

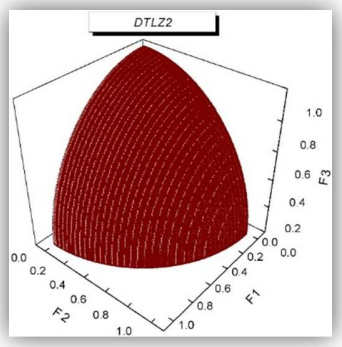


Fig. 4.8 DTLZ2

Minimizar:

$$f_1(x) = \left(1 + g(x_M)\right) cos\left(x_1 \frac{\pi}{2}\right) cos\left(x_2 \frac{\pi}{2}\right) \dots cos\left(x_{M-2} \frac{\pi}{2}\right) cos\left(x_{M-1} \frac{\pi}{2}\right),$$

$$f_2(x) = \left(1 + g(x_M)\right) cos\left(x_1 \frac{\pi}{2}\right) cos\left(x_2 \frac{\pi}{2}\right) \dots cos\left(x_{M-2} \frac{\pi}{2}\right) sin\left(x_{M-1} \frac{\pi}{2}\right),$$

$$f_2(x) = \left(1 + g(x_M)\right) cos\left(x_1 \frac{\pi}{2}\right) cos\left(x_2 \frac{\pi}{2}\right) \dots \ sin\left(x_{M-2} \frac{\pi}{2}\right),$$

…

$$f_{M-1}(x) = \left(1 + g(x_M)\right) cos\left(x_1 \frac{\pi}{2}\right) sin\left(x_2 \frac{\pi}{2}\right),$$

$$f_M(x) = \left(1 + g(x_M)\right) sin\left(x_1 \frac{\pi}{2}\right),$$

Sujeto a: $0 \leq x_i \leq 1, \forall \mathrm{i} = 1, 2, 3, \dots, n$

En donde: $g(x_M) = \sum_{x_i \in X_M} (x_i - 0.5)^2$

Las soluciones Pareto óptimas corresponden a $x_i = 0.5$ para todos las $x_i \in x_M$ y los valores de la función objetivo deben satisfacer $\sum_{m=1}^{M} (f_i)^2 = 1$, el número de variables está dada por $n = M + k - 1$, en donde $k = 10$ como se sugiere en [Coello C. et. al, 2007].

DTLZ3

Este problema es muy similar al DTLZ2, la diferencia radica en la función $g(x_M)$ [Coello C. et. al, 2007]:

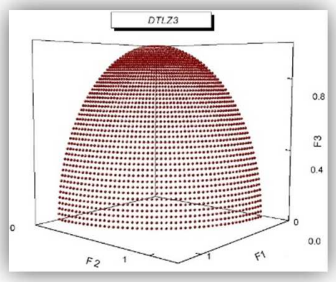


Fig. 4.9 DTLZ3

Minimizar:

$$f_1(x) = \left(1 + g(x_M)\right) cos\left(x_1 \frac{\pi}{2}\right) cos\left(x_2 \frac{\pi}{2}\right) \dots cos\left(x_{M-2} \frac{\pi}{2}\right) cos\left(x_{M-1} \frac{\pi}{2}\right),$$

$$f_2(x) = \left(1 + g(x_M)\right) cos\left(x_1 \frac{\pi}{2}\right) cos\left(x_2 \frac{\pi}{2}\right) \dots cos\left(x_{M-2} \frac{\pi}{2}\right) sin\left(x_{M-1} \frac{\pi}{2}\right),$$

$$f_2(x) = \left(1 + g(x_M)\right) cos\left(x_1 \frac{\pi}{2}\right) cos\left(x_2 \frac{\pi}{2}\right) \dots \ sin\left(x_{M-2} \frac{\pi}{2}\right),$$

…

$$f_{M-1}(x) = \left(1 + g(x_M)\right) cos\left(x_1 \frac{\pi}{2}\right) sin\left(x_2 \frac{\pi}{2}\right),$$

$$f_M(x) = \left(1 + g(x_M)\right) sin\left(x_1 \frac{\pi}{2}\right),$$

Sujeto a: $0 \leq x_i \leq 1, \forall i = 1, 2, 3, \dots, n$

En donde: $g(x_M) = 100\left[|x_M| + \sum_{x_i \in x_M}(x_i - 0.5)^2 - cos(20\pi(x_i - 0.5))\right]$

Este problema cuenta con $(3k - 1)$ frentes de Pareto óptimos locales y un frente de Pareto óptimo global, todos los frentes óptimos locales son paralelos al frente óptimo global, dándose el óptimo global cuando $g^* = 0$, siendo $x_M = (0.5, 0.5, \dots, 0.5)^T$, el siguiente óptimo local se obtiene para $g^* = 1$. El número de variables está dada por $n = M + k - 1$, para este experimento se tomó el valor de $k = 10$ que se sugiere en [Coello C. et. al, 2007].

DTLZ7

Posee un frente de Pareto verdadero desconectado [Coello C. et. al, 2007]:

Minimizar:

$f_1(x) = x_1,$

$f_2(x) = x_2,$

…

$f_{M-1}(x) = x_{M-1},$

$f_M(x) = \left(1 + g(x_M)\right) . h(f_1, f_2, \ldots, f_{M-1}, g(x))$

Sujeto a: $0 \leq x_i \leq 1, \forall i = 1, 2, 3, \ldots, n$

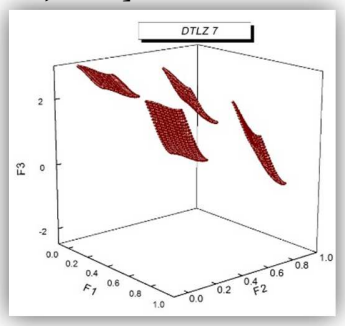


Fig. 4.10 DTLZ7

En donde: $g(x_M) = 1 + \frac{9}{|x_M|}\sum_{x_i \in x_M} x_i,$

$$h(f_1, f_2, \ldots, f_{M-1}, g) = M - \sum_{i=1}^{M-1}\left(\frac{f_i}{1+g(x)}(1 + sin(3\pi f_i))\right)$$

Este problema tiene $2M - 1$ regiones Pareto óptimas desconectadas en el espacio de búsqueda. La función g requiere $k = |x_M|$ variables de decisión, el número total de variables requerido es de $n = M + k - 1$. Este experimento se realizó con $k = 20$. Las soluciones Pareto óptimas corresponden a $x_M = 0$ [Coello C. et. al, 2007].

4.5.3.2 MSS-GP y A^2-NSGA-III utilizando las instancias del problema Selección de Cartera de Proyectos

En este experimento las metaheurísticas MSS-GP y A^2-NSGA-III son evaluadas bajo las condiciones descritas en la sección 4.5.3 utilizando las instancias o3p100_1, o9p100_1 y o16p500_1 de tres, nueve y 16 objetivos respectivamente del problema de Selección de Cartera de Proyectos. En este experimento se tomaron en cuenta restricciones de área, región, presupuesto total, umbrales de indiferencia y veto. Las instancias de tres y nueve objetivos contienen 100 proyectos cada una y la de 16 objetivos 500.

4.5.3.3 MSS y MSS-GP utilizando las instancias del problema de Selección de Cartera de Proyectos

Este experimento se llevó a cabo con la intención de comprobar que el algoritmo MSS mejoró con los cambios realizados que lo llevaron a convertirse en MSS-GP, además muestra que tan bueno es con respecto a su antecesor a través de una comparación entre resultados mediante el indicador de desempeño de dominancia. Para esto se utilizaron las tres instancias del problema Selección de Cartera de Proyectos utilizadas en el experimento descrito en 4.5.3.2, es decir, o3p100_1, o9p100_1 y o16p500_1 de tres, nueve y 16 objetivos respectivamente. En este experimento también se consideraron las restricciones de área, región, presupuesto total, umbrales de indiferencia y veto.

Los parámetros iniciales son los descritos en 4.5.3 para las dos metaheurísticas con una excepción, MSS únicamente utiliza la combinación Uniforme.

4.5.3.4 MSS-GP y NO-ACO utilizando las instancias del problema de Selección de Cartera de Proyectos

En este experimento se compararon las soluciones resultantes de las metaheurísticas MSS-GP y NO-ACO, se llevó acabo un ajuste manual de argumentos para que estuvieran en igualdad de condiciones.

Se comparó la calidad de las soluciones obtenidas de MSS-GP con las de NO-ACO debido a que esta última es la metaheurística que proporcionó la mejor calidad en los resultados de todo el estado del arte (capítulo 3).

La comparación se llevó acabo utilizando el indicador de desempeño de Dominancia. Se utilizaron tres instancias del problema Selección de Cartera de Proyectos usadas en 4.5.3.2: o3p100_1, o9p100_1 y o16p500_1 de tres, nueve y 16 objetivos respectivamente. En este experimento también se consideraron las restricciones de área, región, presupuesto total, umbrales de indiferencia y veto.

4.5.4 Resultados y conclusiones

En este apartado se describen los resultados y conclusiones generados a partir de los experimentos evaluados en las secciones 4.5.3.1, 4.5.3.2, 4.5.3.3 y 4.5.3.4.

Las Tablas 4.7, 4.8, 4.9 y 4.10 muestran los resultados de un experimento para evaluar el desempeño y la correctez de los algoritmos MSS y A^2-NSGA-III. Este experimento que consistió en llevar a cabo 30 ejecuciones por cada metaheurística para cada problema descrito en el apartado 4.5.3. Del conjunto de soluciones obtenidas durante el total de ejecuciones de cada una de las metaheurísticas para un problema se aplica un proceso de eliminación de soluciones repetidas. Finalmente, se aplican las métricas de desempeño descritas en la sección 2.9 al conjunto resultante del proceso de eliminación.

Cada una de las tablas de resultados pertenece a un problema de la sección 4.5.3.1 en particular. Las tablas de resultados poseen tres columnas, la primera indica la métrica a la que hacen referencia los números que la acompañan en la misma fila, la segunda columna muestra el resultado obtenido en la métrica que indica la columna uno en el frente 0 de A^2-NSGA-III y la tercer columna contiene el resultado del indicador que corresponde a cada fila (mostrado en la primer columna) con respecto al frente 0 de MSS.

Las tablas tienen sombreadas las celdas que muestran el mejor desempeño para cada par de resultados.

Tabla 4.7. Comparación de métricas sobre resultados en el problema DTLZ1 con A^2-NSGA-III y MSS.

Métrica	A^2-NSGA-III	MSS
Dominancia	0% (0 de 3150)	80% (2103 de 2623)
Distancia generacional	0.2705128345631	0.1648935527606
Spread	0.9976879123899	3.0564906833993
Hipervolumen	0.4097273349662	0.0695275303113
Distancia generacional invertida	0.237909402498	8.9070012705E-4

Tabla 4.8 Comparación de métricas sobre resultados en el problema DTLZ2 con A^2-NSGA-III y MSS.

Métrica	A^2-NSGA-III	MSS
Dominancia	0% (0 de 3150)	84% (2210 de 2625)
Distancia generacional	0.06998420681142	0.03075450271519
Spread	0.99444149185369	1.10403633584492
Hipervolumen	0.81635150386477	0.03670762667634
Distancia generacional invertida	0.02386227314963	0.00156481831104

Tabla 4.9 Comparación de métricas sobre resultados en el problema DTLZ3 con A^2-NSGA-III y MSS.

Métrica	A^2-NSGA-III	MSS
Dominancia	0% (0 de 3150)	67% (1699 de 2503)
Distancia generacional	0.09094320654874	0.10477220005374
Spread	0.89853334577895	1.162603035902688
Hipervolumen	0.27489234901311	0.00382498322743
Distancia generacional invertida	0.06263774619678	0.00361415478152

Tabla 4.10 Comparación de métricas sobre resultados en el problema DTLZ7 con A^2-NSGA-III y MSS.

Métrica	A^2-NSGA-III	MSS
Dominancia	0% (0 de 3150)	7% (199 de 2576)
Distancia generacional	0.09797319187045	0.010386468176148
Spread	0.99923829970796	1.375567803450116
Hipervolumen	0.94529488839585	0.146080522432978
Distancia generacional invertida	0.12991813492293	0.007370628928402

Conclusión. En esta evaluación hemos se presentó una variante de la metaheurística MSS, que tiene como particularidad la combinación de dos estrategias de mejora para lograr un mejor balance entre calidad y diversidad del frente 0 obtenido es aplicada a varios problemas pertenecientes a la familia DTLZ, se pudo demostrar a través de la evaluación de cinco diferentes métricas que el algoritmo MSS muestra un mejor desempeño comparándolo con el de la metaheurística A^2-NSGA-III.

Como se observa en las tablas 4.7 – 4.10, la calidad de las soluciones de MSS es superior a las de A^2-NSGA-III, tal como lo muestra la métrica de dominancia en cada tabla. La GD y la IGD indican que las soluciones compromiso devueltas por MSS son bastante más cercanas al frente óptimo de Pareto. El hipervolumen se muestra equilibrado ya que cada metaheurística supero a la otra en dos problemas. La técnica Spread sin embargo, muestra que A^2-NSGA-III es mejor que MSS en lo que se refiere a una mejor distribución de las soluciones.

A continuación se presentan las tablas comparativas resultantes del experimento para evaluar el desempeño de las metaheurísticas MSS-GP y A^2-NSGA-III sobre las instancias del problema de Selección de Cartera de Proyectos. Para esta evaluación se utilizó el indicador de desempeño de dominancia descrito en el apartado 2.9 para el problema detallado en el apartado 4.5.3.2, las soluciones comparadas fueron el resultado de 30 corridas y un filtrado final para eliminar las soluciones repetidas:

Tabla 4.11 Comparación mediante el indicador de desempeño de Dominancia sobre las soluciones devueltas por los algoritmos A^2-NSGA-III y MSS-GP para las instancias del problema de Selección de Cartera de Proyectos.

No. Objetivos	A^2-NSGA-III	MSS-GP
3	14% (469 de 3146)	19% (422 de 2149)
9	10% (459 de 4455)	81% (1863 de 2275)
16	0% (0 de 4080)	58% (1393 de 2370)

Conclusión. Se utilizó una variante del MSS-GP en la que se introdujo el MPDF y una S_r para guiar la búsqueda hacia soluciones acorde a los criterios del DM. Al comparar los resultados de este algoritmo con los obtenidos de A^2-NSGA-III se llega a la conclusión que MSS-GP supera en calidad a los resultados de las soluciones de A^2-NSGA-III para las instancias de tres, nueve y 16 objetivos utilizadas, tal como se puede observar en la Tabla 4.11.

Para el tercer experimento se implementó un cambio con el código de MSS para que estuviera en igualdad de condiciones con MSS-GP, se utilizaron las instancias del problema de Selección de Cartera de Proyectos y se implementó el MPDF al final de la última generación en cada una de las 30 corridas. Los resultados después de comparar las 30 corridas

después de aplicar un filtro de no repetición de soluciones (a través de un código también desarrollado en Java) para cada una de las instancias se presentan en la tabla 4.12:

Tabla 4.12 Comparación MSS vs MSS-GP

#Obj.	MSS	MSS-GP
3	76% (38 de 50)	68% (50 de 73)
9	95% (43 de 45)	98% (72 de 73)
16	0% (0 de 58)	78% (136 de 174)

Conclusiones. Como se muestra en la tabla 4.12, con la instancia de tres objetivos el algoritmo MSS muestra un mejor desempeño que el mostrado por MSS-GP, esto es debido a que la cantidad de iteraciones efectuadas por este algoritmo es directamente proporcional al número de objetivos de la instancia utilizada, es decir, a mayor número de objetivos mayor número de iteraciones, debido a que la guía por preferencias implementada por MSS-GP se ejecuta dos veces por iteración, entonces a menor número de iteraciones existe un menor impacto de esta estrategia, es por esto que con la instancia de nueve objetivos ya produce mejores resultados que MSS y con la instancia de 16 objetivos también lo supera a MSS pero con una diferencia de resultados más amplia.

Finalmente la experimentación efectuada en 4.5.3.4 mostró después de comparar las 30 corridas después de aplicar el filtro de no repetición de soluciones para cada una de las instancias se presentan en la tabla 4.13:

Tabla 4.13 Comparación MSS-GP vs NO-ACO

#Obj.	MSS-GP	NO-ACO
3	76% (61 de 80)	100% (33 de 33)
9	100% (45 de 45)	100% (9 de 9)
16	100% (174 de 174)	-% (0 de 0)

Conclusiones. En la tabla 4.13, con la instancia de tres objetivos el algoritmo NO-ACO muestra un mejor desempeño que el mostrado por MSS-GP, aunque también se puede observar como en la cantidad de soluciones finales MSS-GP duplica a la del NO-ACO, la diferencia se debe a que NO-ACO en la mayoría de las corridas llegó a las mismas soluciones finales, por tanto, de 30 corridas solo fueron encontradas 33 soluciones distintas, a diferencia de las 80 obtenidas con MSS-GP. Para la instancia de nueve objetivos los porcentajes de soluciones no dominadas fueron iguales, sin embargo la diferencia del número de soluciones es aún más grande, mientras que la cantidad de soluciones presentadas por MSS-GP aumentó, las de NO-ACO disminuyeron debido a que hubo un incremento en las repeticiones, d 30 corridas solo nueve soluciones distintas fueron encontradas. Finalmente para la instancia de 16 objetivos NO-ACO presentó un problema de configuración lógica debido a que después de cinco días de ejecución no mostró errores pero tampoco pudo finalizar alguna de las 12 ejecuciones que se realizaron en paralelo, mientras que MSS-GP proporcionó 174 soluciones distintas con el mismo nivel de calidad. Por tanto se concluye que a mayor número de objetivos el algoritmo MSS-GP muestra un mejor desempeño que NO-ACO.

CAPÍTULO 5

Análisis de resultados y conclusiones

En el presente capítulo se presentan las aportaciones generadas por la presente investigación, además de los trabajos futuros que se sugiere realizar para alcanzar mejores soluciones sobre el problema de Selección de Cartera de Proyectos.

5.1 Conclusiones

Este trabajo de investigación presenta el desarrollo metódico de dos metaheurísticas para muchos objetivos que brindan solución al problema de Selección de Cartera de Proyectos, también muestra una interesante propuesta de modificación de uno de ellos, dicha propuesta superó ampliamente a todas las metaheurísticas tratadas en el capítulo 4, así como a metaheurísticas del estado del arte.

Las metaheurísticas requieren de pruebas empíricas que muestren el comportamiento y la calidad de las respuestas obtenidas. En este trabajo se evaluaron las metaheurísticas A^2-NSGA-III y MSS, inicialmente se observó el proceso de búsqueda que estuvo acorde a lo planteado en el diseño de la heurística, mostrando que el algoritmo planteado cumplía con lo propuesto. Seguidamente se probó su funcionamiento sobre problemas estándares de la familia DTLZ con lo que se determinó la correctez de sus resultados. Una vez seguros de que los resultados eran factibles y correctos se procedió a comparar las metaheurísticas con las más destacadas dentro del estado del arte, los resultados mostraron que tanto A^2-NSGA-III como MSS tienen un desempeño similar, de los resultados obtenidos se puede destacar la buena dispersión en los resultados obtenidos con A^2-NSGA-III además de su calidad, por otro lado MSS muestra una dispersión no tan uniforme pero la calidad de sus resultados y velocidad

de ejecución son destacables. La calidad de ambas metaheurísticas fue evaluada utilizando algunos problemas estándar pertenecientes a la familia DTLZ mediante las métricas de calidad Spread, Hipervolumen, Distancia Generacional, Distancia Generacional Invertida, así como el indicador de calidad de Dominancia, los resultados se resumen en la tabla 5.1.

Tabla 5.1 Comparación de calidad entre A^2-NSGA-III y MSS

Algoritmo		**A^2-NSGA-III**	**MSS**
DTLZ 1	*Hipervolumen*	**0.41**	0.07
	Spread	**0.99**	3.05
	Distancia Generacional	0.27	**0.16**
	Distancia Generacional Invertida	0.24	**8.9e-4**
	Dominancia	0%	**80%**
DTLZ 2	*Hipervolumen*	**0.82**	0.04
	Spread	**0.99**	1.1
	Distancia Generacional	0.07	**0.03**
	Distancia Generacional Invertida	0.02	**0.001**
	Dominancia	0%	**84%**
DTLZ 3	*Hipervolumen*	**0.27**	0.003
	Spread	**0.9**	1.16
	Distancia Generacional	**0.09**	0.104
	Distancia Generacional Invertida	0.06	**0.003**
	Dominancia	0%	**67%**
DTLZ 7	*Hipervolumen*	**0.95**	0.15
	Spread	**0.99**	1.37
	Distancia Generacional	0.98	**0.01**
	Distancia Generacional Invertida	0.13	**0.007**
	Dominancia	0%	**7%**

Como se observa, los resultados de las métricas que se enfocan en la dispersión y uniformidad de las soluciones sobre el espacio de búsqueda son dominados por A^2-NSGA-III, mientras que en las métricas que se enfocan en la calidad de las soluciones por el contrario destacan a la metaheurística MSS.

Se experimentó con la incorporación de preferencias a las dos metaheurísticas propuestas en este trabajo y se concluyó que incorporan una mejora a la calidad de las soluciones, al mismo tiempo restan dispersión en las mismas.

Uno de los puntos más destacables en este trabajo es la incorporación de una nueva estrategia apoyada en el uso del modelo de preferencias difusas, esta estrategia utiliza la característica del modelo de preferencias para enfocar la búsqueda hacia la región de interés del DM, lo cual

permite una exploración dirigida hacia una región reducida del espacio de búsqueda teniendo por consecuencia un incremento en la calidad de las soluciones. Debido a que MSS mostró una mejor calidad en las soluciones que A^2-NSGA-III y estuvo al nivel de calidad de las mejores metaheurísticas en el estado del arte, se decidió incorporar esta nueva estrategia con la intención de aprovechar el buen desempeño en calidad de las soluciones, buscando un nuevo algoritmo que sobresaliera en ese aspecto sobre el resto del estado del arte, a este algoritmo se le llamó MSS-GP y los resultados muestran que MSS-GP supera en dominancia a los algoritmos ACO, NSGA-II, NSGA-III, A-NSGA-III, A^2-NSGA-III y el mismo MSS con los que se comparó.

Finalmente se puede concluir que los objetivos planteados en la sección 1.3 fueron cumplidos satisfactoriamente: se diseñaron e implementaron las metaheurísticas MSS y A^2-NSGA-III para dar solución al problema de Selección de Cartera de Proyectos; se realizó un estudio experimental con el fin de ajustar los parámetros de ambas metaheurísticas; se llevó a cabo un estudio apoyado en métricas e indicadores de desempeño sobre las soluciones de ambas metaheurísticas para comparar su calidad y diversidad, concluyendo que ambas metaheurísticas proporcionan soluciones competitivas con las metaheurísticas del estado del arte, además se determinó que la estrategia de diversidad de A^2-NSGA-III tiene un mejor desempeño que el de la estrategia de diversidad de MSS, sin embargo MSS proporciona soluciones con una mayor calidad.

Además como un extra se creó una estrategia que guía apoyándose en preferencias difusas que se implementó dentro de MSS dando como resultado el MSS-GP que superó las expectativas al proporcionar mejor calidad en sus resultados que todas las metaheurísticas del estado del arte con las que se comparó.

5.2 Trabajos futuros

Una vez estipulada la variante de MSS (MSS-GP), y establecido su debido algoritmo, el principal trabajo futuro es la incorporación de una variante similar basada en preferencias a otras metaheurísticas, debido al buen desempeño mostrado en esta investigación.

Referencias bibliográficas

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

[Bastiani S., 2013] Bastiani, S.: Solución de problemas de Cartera de Proyectos Públicos a partir de información del ranking de prioridades (2013).

[Coello C. et. al 2007] Coello, C., Lamont, B., Van, D.: Evolutionary algorithms for solving multi-objective problems. Genetic and evolutionary computation series, 2a. edición. pp. 200-206 (2007).

[Coello C., 2013] Coello, C.: Introducción a la computación evolutiva. CINVESTAV-IPN (2013).

[Colomine F. et. al, 2012] Colomine, F., Carlos, C., Fernández, A.: A comparative study of multi-objective evolutionary algorithms to optimize the selection of investment portfolios with cardinality constraints. EvoApplications 165-173 (2012).

[Cruz-Reyes L. et al., 2013a] Cruz-Reyes, L., Fernández, E., Olmedo, R., Sánchez, P., Navarro, J.: Preference Incorporation into evolutionary multiobjective optimization using preference information implicit in a set of assigment examples, Proceedings of fourth international workshop on knowledge discovery, knowledge management and decisión support . Atlantis press (2013).

[Cruz-Reyes L. et al., 2013b] Cruz-Reyes, L., Fernández, E., Gómez, C., Rivera, G.: Multicriteria optimization of interdependent project portfolios with ‘a priori’ incorporation of decision maker preferences. Eureka-2013. Fourth International Workshop Proceedings, Published by Atlantis Press, 169-178 (2013).

[Cruz-Reyes L. et al., 2014] Cruz-Reyes, L., Fernandez, E., Gómez, C. y Sanchez, P.: Preference incorporation into evolutionary multiobjective optimization using a multi-criteria evaluation method. Recent advances on hybrid approaches for designing intelligent systems, pp. 533-542 (2014).

[Davis R. y McKeown P., 1984] Davis, R. y McKeown P.: Modelos cuantitativos para administración. Grupo editorial Iberoamérica (1984).

[Deb K. et al., 2002] Deb, K., Agrawal, S., Pratap, A. and T. Meyarivan: A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans. Evolutionary Computation 6(2): 182-197 (2002)

[Deb K. y Jain H., 2013] Deb, K., Jain, H.: An evolutionary many-objective optimization algorithm using reference-point based non-dominated sorting approach, part II: Handling constraints and extending to an adaptive approach. IEEE Transactions on Evolutionary Computation 18(4), pp. 602-622 (2013).

[Deb K. y Jain H., 2014] Deb, K., Jain, H.: An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part I: solving problems with box constraints. IEEE trans. Evol. Comput. 18(4), pp. 577-601 (2014).

[Fernández E. et al., 2013] Fernández, E., López, E., Mazcorro, G., Olmedo, R. y Coello C.: Application of the non-Outranked Sorting Genetic Algorithm to public project portfolio selection. Inf Sci, 228:131–149 (2013).

[García L. y Muñoz A., 2009] García, C. y Muñoz, P.: Localización empresarial en Aragón: Una aplicación empírica de la ayuda a la decisión multicriterio tipo ELECTRE I y III. Robustez de los resultados obtenidos, Revista de Métodos Cuantitativos para la Economía y la Empresa 7, pp. 31-56 (2009).

[Gento A. y Redondo A., 2005] Gento, A. y Redondo, A., Comparación del método Electre III y Promethee II: Aplicación al caso de un automóvil. IX Congreso de ingeniería de organización Gijón (2005).

[Glover F., 1977] Glover, F.: Heuristics for integer programming using surrogate constraints. Decision Sciences, Vol. 8, No. 1, pp. 156–166 (1977).

[Jain H. y Deb K., 2013] Jain, H., Deb, K.: An Improved Adaptive Approach for Elitist Nondominated Sorting Genetic Algorithm for Many-Objective Optimization. EMO, pp. 307-321 (2013).

[Lazarini L. y López J., 2009] Lanzarini, L. y López, J.: Metaheurísticas poblacionales aplicadas a la resolución de problemas complejos. Instituto de investigación en informática LIDI, facultad de informática UNLP (2009).

[Martí R. y Laguna M., 2003] Martí, R. y Laguna, M.: Scatter Search: Diseño básico y estrategias avanzadas. Inteligencia artificial. Revista Iberoamericana de inteligencia artificial, p. 19 (2003).

[Melián B. et al., 2003] Melián, B., Moreno, J., y Moreno M.: Metaheuristicas: Una visión global. Inteligencia Artificial, Revista Iberoamericana de Inteligencia Artificial 7(19): 7-28 (2003).

[Mobasher B. et al., 2004] Bamshad, M., Xin, J., Yanzan, Z.: Semantically enhanced collaborative filtering on the web in web mining: from web to semantic web, Vol. 3209/2004. Springer Berlin / Heidelberg, pp. 57-76 (2004).

[Molica F. et al., 2011] Molica, F., Durange, C. y Bastos, R.: Aplicación del método Electre III en la clasificación de clústeres de artesanías. Revista INGE CUC, Vol. 7, No. 1. ISSN 0122-6517, pp. 97-112 (2011).

[Nebro A. et al., 2007] Nebro, A., Luna, F., Dorronsoro, B. y Durillo, J.: Un algoritmo multiobjetivo basado en búsqueda dispersa. Quinto Congreso Español de Metaheurísticas, Algoritmos Evolutivos y Bioinspirados (MAEB 2007), pp. 175-182 (2007).

[Nebro A. et al., 2008] Nebro, A., Luna, F., Alba, E., Dorronsoro, B., Durillo, J. y Beham, A.: AbYSS: Adapting scatter search to multiobjective optimization. IEEE Trans. evolutionary computation, vol. 12 No. 4, pp. 439-457 (2008).

[Osyczka A., 1985] Osyczka, A.: Multicriteria optimization for engineering design. Academic press, Cambridge, MA, pp. 193–227 (1985).

[Öztürk M. et al., 2005] Öztürk, M., Tsoukiàs, A. y Vincke, P.: Preference modelling. Preferences (2005).

[Rodríguez-Piñero T., 2003] Rodríguez-Piñero T.: Introducción a los algoritmos genéticos y sus aplicaciones. Universidad Rey Juan Carlos, España, Madrid (2003).

[Roy B., 1990] Roy, B.: Reading in multiple criteria decision aid, chapter the outranking approach and the foundations of ELECTRE methods. Springer-Verlag, pp.155-183 (1990).

[Roy B., 1996] Roy, B.: Multicriteria methodology for decision aiding. Kluwer academic publishers, Dordrecht (1996).

[Roy B. y Slowinski, 2008] Roy, B. y Slowinski, R.: Handling effects of reinforced preference and counter-veto in credibility of outranking. European journal of operational research, Vol. 188, No. 1, pp. 185-190 (2008).

[Sánchez P., 2012] Sánchez, P.: Propuesta de anteproyecto de tesis: Nuevos métodos de incorporación de preferencias en metaheurísticas multiobjetivo para la solución de problemas de cartera de proyectos (2012).

[Smith Q. et al., 2000] Smith, Q., Mesa, S., Dyner, R., Jaramillo, A., Poveda, J. y Valencia R.: Decisiones con múltiples objetivos e incertidumbre. 2 Ed. R. Medellín: Universidad Nacional de Colombia (sede Medellín). Facultad de Minas, p. 354 (2000).

[Srinivas N. y Deb K., 1995] Srinivas, N. y Deb, K.: Multiobjective function optimization using nondominated sorting genetic algorithms. Evol. comput., Vol. 2, No. 3, pp. 221–248 (1995).

[Tapabrata R. et al., 2013] Tapabrata, R., Asafuddoula, Md. e Isaacs, Amitay: A steady state decomposition based quantum genetic algorithm for many objective optimization. IEEE Congress on evolutionary computation, pp. 2817-2824 (2013).

[Zitzler E. et al., 2001] Zitzler, E., Laumanns, M. y Thiele, L.: SPEA2: Improving the strength pareto evolutionary algorithm, Tech. Rep. 103, Swiss Federal Institute of Technology (ETH), Zurich, Switzerland (2001).