Centro de Investigación y de Estudios Avanzados
del Instituto Politécnico Nacional

Unidad Zacatenco

Departamento de Computación

Un Nuevo Optimizador Multi-Objetivo Mediante Cúmulos de Partículas Basado en SPSO2011

Tesis que presenta

Juan Luis Salazar Mendoza

para obtener el Grado de

Maestro en Ciencias en Computación

Director de la Tesis

Dr. Carlos Artemio Coello Coello

Ciudad de México Diciembre 2016

Resumen

En el mundo real, la mayor parte de los problemas de optimización tienen dos o más objetivos (normalmente en conflicto) que deseamos satisfacer al mismo tiempo. Este tipo de problemas, llamados multi-objetivo, han sido abordados con diversas técnicas de programación matemática. Sin embargo, no existe ningún método de optimización que pueda garantizar la solución de cualquier problema multi-objetivo no lineal, lo que abre la puerta al uso de métodos alternativos de optimización.

En los últimos 20 años, las metaheurísticas bio-inspiradas (sobre todo, los algoritmos evolutivos) han sido ampliamente utilizadas para resolver problemas de optimización multi-objetivo no lineales, debido, entre otras cosas, a su generalidad, sus pocos requerimientos de información y su uso de una población, que les permite generar varias soluciones no dominadas en una sola ejecución algorítmica.

La optimización mediante cúmulos de partículas, es un tipo de metaheurística bio-inspirada que simula el comportamiento de un grupo de aves que buscan comida. Este tipo de técnica ha mostrado ser un optimizador mono-objetivo muy eficiente, lo cual constituye una motivación importante para intentar extenderla a la solución de problemas multi-objetivo.

En este trabajo de tesis se propone un nuevo algoritmo de optimización mediante cúmulos de partículas multi-objetivo utilizando los operadores del estándar de 2011 de esta metaheurística, llamado SPSO2011 [1]. Ésta es la versión mono-objetivo más competitiva que se conoce a la fecha del algoritmo de optimización mediante cúmulos de partículas, lo que la hace ideal para extenderla a la solución de problemas multi-objetivo. El algoritmo propuesto se basa en un método de descomposición, a fin de explotar de mejor manera los operadores del SPSO2011. El desempeño del algoritmo propuesto se compara con el de algoritmos evolutivos multi-objetivo del estado del arte (MOEA/D)[2], y con el de optimizadores multi-objetivo basados en cúmulos de partículas (SMPSO [3] y dMOPSO [4]), usando problemas de prueba e indicadores de desempeño comúnmente utilizados en la literatura especializada.

Abstract

In the real world, most optimization problems have two or more (normally conflicting) objectives that we aim to satisfy at the same time. This type of problems, called multi-objective, has been tackled using different mathematical programming techniques. However, there is no optimization method that can guarantee the solution of every nonlinear multi-objective optimization problem, which opens the door to the use of alternative optimization problems.

In the last 20 years, bio-inspired metaheuristics (particularly, evolutionary algorithms) have been widely used to solve nonlinear multi-objective optimization problems, due to, among other things, their generality, their few information requirements, and their use of a population, which allows them to generate several nondominated solutions in a single algorithmic execution.

Particle swarm optimization is a type of bio-inspired metaheuristic that simulates the behavior of a group of birds seeking for food. This sort of technique has been shown to be a very efficient single-objective optimizer, which constitutes an important motivation to try to extend it to the solution of multi-objective problems.

In this thesis work, we propose a new multi-objective particle swarm optimizer using the operators of the 2011 standard of this metaheuristic, called SPSO2011 [1]. This is the most competitive single-objective version known to date of particle swarm optimization, which makes it ideal to extend it to solve multi-objective problems. The proposed algorithm is based on decomposition, in order to better exploit the operators of SPSO2011. The performance of the proposed algorithm is compared with respect to that of state-of-the-art multi-objective evolutionary algorithms (MOEA/D) [2], and multi-objective particle swarm optimizers (SMPSO [3] and dMOPSO [4]) using standard test problems and performance indicators taken from the specialized literature.

Agradecimientos

Quiero agradecer a mi hija Ximena, quien fue mi motor y principal motivación para ser cada día mejor durante este proceso de aprendizaje y crecimiento tanto académico como personal.

Quiero agradecer a mi madre Maria Elena, a mis hermanos Daniel, Maria Luisa, Rogelio, gracias por apoyarme en esta etapa de mi vida, con sus consejos, con su paciencia, con su tiempo y sobre todo con su amor incondicional.

A mis compañeros de la generación 2014, por estar siempre unidos en este proceso, grandes personas con los que compartí y aprendí muchísimo.

A mi asesor, el Dr. Carlos Artemio Coello Coello por su tiempo, por compartir tanto conocimiento valioso, por creer en mi, por apoyarme y por mostrarme a través de su ejemplo como ser una gran persona.

A CONACYT por la beca de maestría, ya que sin ella hubiese sido imposible concluir con este sueño.

Este trabajo de tesis se derivó del proyecto titulado "Nuevos Paradigmas Algorítmicos en Optimización Evolutiva Multi-Objetivo" (Ref. 221551) cuyo responsable es el Dr. Carlos A. Coello Coello.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

Cada tipo de metaheurística bio-inspirada constituye un área de investigación en sí y actualmente existe un gran número de variantes de cada una de ellas, incluyendo extensiones multi-objetivo a varias de ellas. El algoritmo de nuestros interés para fines de esta tesis es el de optimización mediante cúmulos de partículas (*particle swarm optimization* o PSO), el cual fue propuesto inicialmente en 1995 [5]. Los PSOs operan sobre poblaciones, lo cual les ayuda a no quedar atrapados en óptimos locales. Además, no necesitan información específica del problema, lo cual los hace adecuados para atacar problemas de optimización global. A la fecha existen muchísimas variantes del PSO para optimización global, entre las cuales destacan las versiones estandarizadas [6], que intentan presentar un punto de referencia para los futuros avances del PSO.

En este trabajo de tesis se propone un PSO para optimización multi-objetivo (o sea un *multi-objective particle swarm optimizer* o MOPSO) que se base en el PSO estándar del 2011 (SPSO2011) ya que hasta ahora, este algoritmo no se ha extendido a problemas multi-objetivo. La razón para elegir el SPSO2011 es que este algoritmo ha demostrado ser muy bueno en optimización global [1] y de ahí nuestro interés en extenderlo a problemas multi-objetivo.

1.1. Objetivos

El objetivo general de esta tesis es proponer un MOPSO basado en el SPSO2011, que logre resolver adecuadamente problemas de optimización multi-objetivo no lineales que tengan de uno a diez objetivos.

particulares

1. Diseñar un MOPSO que introduzca los operadores del SPSO2011.
2. Definir los alcances y limitaciones del algoritmo.

3. Compararlo con algoritmos evolutivos multi-objetivo del estado del arte tales como *Multi-Objective Evolutionary Algorithm Based on Decomposition* (MOEA/D) [2], *Speed contrained Multi-objective PSO* (SMPSO) [3] y *Decomposition-Based Multi-Objective Particle Swarm Optimizer* (dMOPSO) [4].

1.2. Estructura del documento

Este documento consta de seis capítulos, que describen el trabajo realizado y los resultados obtenidos, los cuales se describen a continuación:

En el Capítulo 1 se presenta la introducción del trabajo de tesis.

En el Capítulo 2 se describen los conceptos básicos, abarcando la computación evolutiva, y la optimización multi-objetivo y explicando el funcionamiento de la optimización mediante cúmulos de partículas.

En el Capítulo 3 se describe en qué consiste la optimización evolutiva con muchos objetivos, sus principales dificultades y los enfoques principales que abordan los algoritmos propuestos en la literatura, para resolverlos.

En el Capítulo 4 se presenta a detalle el diseño y los componentes del algoritmo propuesto.

En el Capítulo 5 se presenta el diseño experimental para evaluar el algoritmo propuesto, así también como los resultados de dicha experimentación y la comparativa con algoritmos del estado el arte.

En el Capítulo 6 se presentan las conclusiones y el trabajo a futuro, partiendo de los resultados de la experimentación y el proceso de desarrollo de la propuesta.

Capítulo 2

Conceptos básicos

2.1. Computación evolutiva

La Computación Evolutiva es una rama de la inteligencia artificial inspirada en los mecanismos de la evolución y la selección natural. Ésta concibe a la evolución como un proceso de optimización el cual es simulado para resolver problemas. La idea fundamental de la computación evolutiva es simular la evolución de una población de posibles soluciones a un problema, utilizando operadores inspirados en la variación genética y en la selección natural (reproducción, mutación, competencia y selección). Esta simulación de la evolución opera como un método de búsqueda en un espacio de soluciones posibles. El conjunto de opciones de búsqueda consiste en las posibles secuencias genéticas, y las soluciones deseables son los organismos más aptos para sobrevivir y reproducirse en su ambiente.

Conceptos básicos de computación evolutiva

En las técnicas de la computación evolutiva existe una población conformada por individuos los cuales tienen la capacidad de reproducirse con una determinada probabilidad y que representan posibles soluciones al problema que se desea resolver. Al reproducirse los individuos, de manera inherente se realiza un proceso de exploración en el espacio de búsqueda para mejorar cada vez más las posibles soluciones y con esto acercarnos a la solución óptima del problema. A cada individuo se le asocia una aptitud, la cual describe la habilidad o capacidad del individuo; con dicha aptitud se evalúa la supervivencia de un individuo en cada generación. Esto se relaciona con el valor de la función objetivo para la correspondiente interpretación de éste en términos de soluciones del problema. La población se renueva generación tras generación bajo el principio de la supervivencia del más apto. Además, cuenta con diversidad entre sus individuos, con lo cual se busca obtener un muestreo representativo del espacio de búsqueda del problema, es decir, contar con un mecanismo de exploración. Por esta razón, todos los algoritmos evolutivos tienen tres características importantes que los distinguen de otros algoritmos de búsqueda. En primer lugar, son algoritmos poblacionales. En segundo lugar, hay comunicación e intercambio de información entre todos los individuos de la población, mediante la selección o

la recombinación de los individuos. Por último, cuentan con un elemento muy importante: sus operadores son estocásticos. Esto les permite evitar, con mayor facilidad, el quedar atrapados en óptimos locales.

Entre los problemas apropiados para resolver con estas técnicas se encuentran aquellos en los que la región factible no se puede modelar de manera sencilla, así como aquellos en que no se cumplen condiciones de convexidad o conectividad, o cuando no se cuenta con conocimiento previo de las características del espacio de búsqueda. A continuación se definen algunos conceptos básicos de la computación evolutiva:

- **Población:** es el conjunto de individuos que representan las posibles soluciones a un problema. Estos individuos están compuestos de un cromosoma o genoma cuya codificación es llamada genotipo. De manera opuesta se encuentra la decodificación de dicho cromosoma a la cual se denomina fenotipo.

- **Operadores:** son los procedimientos que manipulan la información genética de la población para crear nuevos individuos a partir de la población original. Los operadores genéticos más comunes en los algoritmos evolutivos (AEs) son la cruza y la mutación.

- **Mutación:** es la perturbación sobre uno o más elementos (también denominados como alelos) del cromosoma de un individuo. Permite alcanzar una región del espacio de búsqueda diferente a la del individuo original, lo que ayuda a que la búsqueda no se estanque en un óptimo local.

- **Cruza:** tiene la finalidad de heredar la información genética de dos o más padres a los individuos de la siguiente generación. En la computación evolutiva, la cruza entre cromosomas se simula intercambiando segmentos de cadenas lineales de longitud fija. Los genes de los padres se combinan para formar las cadenas cromosómicas de sus descendientes, buscando preservar los buenos genes y obtener mejores soluciones al problema. El principio detrás de la cruza es combinar dos o más individuos con características deseadas para producir uno o más individuos que combinen tales cualidades.

- **Selección:** es el proceso mediante el cual se determina qué individuos pasarán a formar parte de la siguiente generación. Cada uno de los individuos cuenta con un valor de aptitud determinado, el cual indica la diferencia entre los individuos de la población. A la determinación de dichos valores se le conoce como evaluación de la función de aptitud y dado que el proceso de selección toma en cuenta esta función, se dice que este procedimiento está basado en el principio de supervivencia del más apto.

2.1.1. Principales paradigmas

La computación evolutiva se conforma por tres principales paradigmas: los algoritmos genéticos (AGs), la programación evolutiva (PE) y las estrategias evolutivas (EE). Dichos algoritmos tienen en común el uso de operadores que simulan la reproducción, la mutación, la competencia, y la selección de individuos dentro de una población. A continuación se da una explicación breve acerca del funcionamiento básico de cada uno de estos paradigmas [7].

Algoritmos genéticos

Los algoritmos genéticos (AGs) fueron propuestos por John H. Holland [8] y originalmente fueron denominados planes reproductivos. John H. Holland fue motivado por su interés en resolver problemas de aprendizaje de máquina. Con los AGs se introducen conceptos existentes en la naturaleza, como el genotipo y fenotipo. El genotipo representa la información genética de un individuo, que ha heredado de sus antecesores y que a su vez transmite a sus descendientes. El fenotipo son las características visibles de cada individuo. En computación evolutiva, el fenotipo es la representación decodificada de una posible solución al problema y el genotipo es la cadena cromosómica que codifica dicha solución. El algoritmo genético opera a nivel de genotipo de las soluciones y enfatiza la importancia de la cruza sexual sobre la mutación, por lo que la cruza es el operador principal de este paradigma el cual utiliza una selección probabilística. La cadena binaria es la representación tradicional de los algoritmos genéticos, y es llamada cromosoma. A cada posición dentro del cromosoma correspondiente a una variable de decisión se le denomina gene y cada uno de sus elementos específicos es llamado alelo. De tal forma, para poder aplicar el algoritmo genético se requiere de los siguientes componentes básicos:

- Una representación de las soluciones al problema.
- Un procedimiento para crear una población de posibles soluciones de manera aleatoria.
- Una función de evaluación que simule el ambiente, clasificando los individuos en términos de su aptitud.
- Operadores genéticos que alteren la composición de los descendientes que se producirán para las siguientes generaciones.
- Los parámetros utilizados por el algoritmo genético tales como: el tamaño de población y la probabilidad de cruza, probabilidad de mutación y número máximo de generaciones, entre otros.

Los AGs han sido aplicados en diversas áreas, como por ejemplo: bases de datos, generación de gramáticas, reconocimiento de patrones, etc. En el algoritmo 1 se muestra un algoritmo genético genérico.

Algoritmo 1: Algoritmo genético genérico

Entrada: Tamaño de población (tampob), número máximo de generaciones (Gmax)
Salida: ConjuntoDeSoluciones

```
inicio
    Generar aleatoriamente la población inicial, G(0)
    t:= 0
    mientras Cierta condición de paro no se satisfaga o t < Gmax hacer
        sea t := t + 1
        calcular la aptitud de cada individuo de G(t)
        seleccionar G1(t) con base en la aptitud de G(t)
        aplicar los operadores genéticos a G1(t) para generar G(t + 1)
    fin
fin
devolver ConjuntoDeSoluciones
```

Estrategias evolutivas

Las estrategias evolutivas (EEs) fueron propuestas por varios investigadores alemanes encabezados por Ingo Rechenberg en 1964 [9]. Son un método de ajustes discretos aleatorios inspirado en el mecanismo de mutación que ocurre en la naturaleza. Por esta razón el operador principal es la mutación y la cruza es un operador secundario en las EEs. La versión original de la EEs, denotada por (1 + 1)-EE, usa un solo padre y produce en cada iteración un solo hijo usando mutación. El mejor de entre los dos (el padre y el hijo) pasará a formar parte de la siguiente generación. En la (1 + 1)- EE, a partir de un padre $x^t = (x^*{}_1, x^*{}_2, ..., x^*{}_n)^T$ se genera un hijo mediante:

$$x_i{}^{t+1} = x_i{}^t + N_i(0, \sigma_i{}^t) \tag{2.1}$$

donde t es la generación en la que se encuentra el algoritmo y $N_i(0, \sigma_i{}^t)$ es un vector de números Gaussianos independientes con media cero y desviación estándar σ_i. De tal forma, la mutación opera como una perturbación entre los individuos. Más tarde, el concepto de población fue propuesto por Rechenberg [10] en las EEs, al crear una estrategia evolutiva llamada (μ+1)-EE, la cual está compuesta de μ padres, creándose un solo hijo a partir de éstos, el cual puede reemplazar al peor padre de la población en una selección extintiva, el algoritmo básico de la (1 + 1)- EE se presenta en el algoritmo 2.

Finalmente, Hans-Paul Schwefel introdujo a las estrategias el uso de múltiples hijos en las denominadas (μ + λ)-EE y (μ,λ)-EE, donde μ padres producen λ hijos. La primera estrategia selecciona μ individuos de la unión de padres e hijos, mientras que la segunda selecciona μ individuos de la población de hijos.

Las EEs han sido empleadas para resolver problemas en áreas tales como: bioquímica, óptica y redes, entre muchas otras [11].

Algoritmo 2: Algoritmo básico de la estrategia evolutiva (1 + 1)-EE

```
1 inicio
2     Generar aleatoriamente el padre inicial
3     mientras Cierta condición no se satisfaga hacer
4         se aplica la mutación (2.1) al padre para generar un hijo
5         seleccionar al mejor entre padre e hijo
6         el individuo seleccionado pasa a ser padre
7     fin
8 fin
9 devolver ConjuntoDeSoluciones
```

Programación evolutiva

La programación evolutiva (PE) fue propuesta por Lawrence J. Fogel [12]. En este paradigma, la inteligencia se ve como un comportamiento adaptativo. En este algoritmo se simula la evolución al nivel de las especies. Utiliza una selección probabilística y no requiere de la cruza u otro tipo de recombinación, puesto que, en la naturaleza, una especie no puede mezclarse con otra y este algoritmo evolutivo simula este principio. Enfatiza los nexos de comportamiento entre padres e hijos, en vez de buscar emular operadores genéticos específicos, como lo hacen los algoritmos genéticos.

Algoritmo 3: Algoritmo básico de la programación evolutiva

```
   Entrada: Tamaño de población (tampob)
   Salida: ConjuntoDeSoluciones
1  inicio
2      Generar aleatoriamente la población inicial
3      mientras Cierta condición no se satisfaga hacer
4          calcular la aptitud de cada hijo
5          seleccionar las soluciones que se mutarán
6          aplicar mutación
7          reemplazar la población actual con los individuos mutados
8      fin
9  fin
10 devolver ConjuntoDeSoluciones
```

La PE ha sido utilizada en diferentes áreas tales como: reconocimiento de patrones, predicción, identificación, control automático y juegos, entre otras [12, 13].

El algoritmo 3 muestra un algoritmo básico de programación evolutiva.

2.2. Optimización multi-objetivo

Los problemas de optimización multi-objetivo (POMs) se encuentran en prácticamente todas las áreas del conocimiento. En los POMs, los objetivos a ser optimizados normalmente se encuentran en conflicto entre sí, lo que implica que no tienen una solución única. En los POMs, se obtiene un conjunto de soluciones que representa los mejores compromisos posibles entre todos los objetivos (es decir, soluciones en las que no es posible mejorar un objetivo sin empeorar otro).

En esta tesis se resuelven problemas de optimización multi-objetivo continuos (o sea, cuyas variables de decisión son números reales) y sin restricciones los cuales se definen de la siguiente manera:

$$\min_{\vec{x}\in\Omega} \quad \vec{F}(\vec{x}) \tag{2.2}$$

donde $\vec{x} = (x_1, ..., x_n)$ es un vector solución en $\Omega \subset \mathbb{R}^n$ y $\vec{F}$ se define como el vector de las funciones objetivo:

$$\vec{F} : \Omega \to \mathbb{R}^k, \quad \vec{F}(\vec{x}) = (f_1(\vec{x}), \ldots, f_k(\vec{x}))^T$$

donde cada $f_i : \mathbb{R}^n \to \mathbb{R}$ es una función continua y sin restricciones.

Para describir el concepto de optimización en el que estamos interesados, se proporcionan a continuación las siguientes definiciones [14]:

Definición 1 (Dominancia de Pareto): Un vector $\vec{u} = (u_1, ...u_k)$, domina a otro vector $\vec{v} = (v_1, ...v_k)$ (denotado por $\vec{u} \preceq \vec{v}$) si y sólo si, $\forall i \in \{1, ..., k\}$ $u_i \leq v_i$ y $\exists i \in \{1, ..., k\}$ $u_i < v_i$.

Definición 2 (Óptimo de Pareto): Sea $\vec{x}^\star \in \Omega$, se dice que $\vec{x}^\star$ es un *óptimo de Pareto*, si y sólo si no existe otra solución $\vec{x} \in \Omega$ tal que $\vec{F}(\vec{x}) \preceq \vec{F}(\vec{x}^\star)$.

Definición 3 (Conjunto de óptimos de Pareto): El *conjunto de óptimos de Pareto*, PS, se define como:

$$PS = \{\vec{x} \in \Omega | \vec{x} \text{ es un óptimo de Pareto}\}$$

y su imagen ($PF = \{\vec{F}(\vec{x}) | \vec{x} \in PS\}$) se denomina *frente óptimo de Pareto.*

Para resolver un POM se espera obtener la mayor cantidad de elementos del conjunto de óptimos de Pareto que sea posible, manteniendo al mismo tiempo una distribución de soluciones que sea lo más uniforme posible a lo largo del frente de Pareto.

2.3. Optimización mediante cúmulos de partículas

James Kennedy y Russell C. Eberhart [5] propusieron en 1995 la heurística denominada optimización mediante cúmulos de partículas (PSO por sus siglas en inglés). La idea básica detrás del algoritmo es usar un conjunto de "partículas" para explorar el paisaje de aptitud de un problema particular, simulando el comportamiento social de una parvada de aves que buscan comida.

Aunque en su versión original fue adoptado para balancear pesos en una red neuronal [15], el PSO se convirtió rápidamente en un optimizador global muy popular principalmente en problemas en los que las variables de decisión son números reales [16, 17].

Desde 2006, se han establecido tres versiones estándar del PSO, nombradas SPSO 2006, 2007, y 2011 [6]. En general, cada una tiene ligeras modificaciones comparadas con su versión anterior. Al ser diseñadas, se utilizaron los últimos avances teóricos hasta ese momento para establecer un algoritmo de referencia en cuanto a las variantes futuras.

A continuación se formalizará la notación necesaria para entender esta heurística y se explicarán las diferencias entre las diferentes versiones.

2.3.1. Notación necesaria

Para poder explicar la heurística es necesario definir la siguiente notación:

w es el peso inercial que lleva la partícula y es empleado para inducir cierto grado de la velocidad anterior en la velocidad actual.

c_1 es el factor de aprendizaje cognitivo, y representa la atracción de la partícula a la mejor dirección de acuerdo a su propia experiencia.

c_2 es el factor de aprendizaje social, y representa la atracción de la partícula hacia la mejor dirección de acuerdo al comportamiento social.

r_1 y r_2 son valores aleatorios que regulan la implicación del factor de aprendizaje cognitivo y social.

D es la dimensionalidad del espacio de búsqueda, es decir, la cantidad de variable de decisión del POM.

E es el espacio de búsqueda, un hiper-paralelepípedo definido como el producto Euclidiano de los D intervalos reales.

$$E = \bigotimes_{d=1}^{D} [\text{mín}_d, \text{máx}_d] \tag{2.3}$$

f es la función de aptitud definida en E.
t es la iteración actual.
$x_i(t)$ es la posición i en la iteración t.
$v_i(t)$ es la velocidad en la iteración t.
$p_i(t)$ es la mejor posición personal hasta la iteración t.
$l_i(t)$ es la mejor posición personal encontrada en el vecindario hasta la iteración t.

2.3.2. Inicialización de cúmulos

Sea $N_i(t)$ el conjunto de vecinos de la partícula i en el tiempo t. Se utiliza una primera topología [18] para conectar a los vecinos en cada vecindario $N_i(0)$. Posteriormente, inicializamos cada partícula de la siguiente manera:

En SPSO 2006 y 2007:

$x_i(0) = U(\text{mín}_d, \text{máx}_d)$
$v_i(0) = \frac{U(\text{mín}_d, \text{máx}_d) - x_i(0)}{2}$
$p_i(0) = x_i(0)$
$l_i(0) = \text{argmin}_{j \in N_i(0))}(f(p_j(0)))$

Donde $U(\text{mín}_d, \text{máx}_d)$ es un número aleatorio entre $[\text{mín}_d, \text{máx}_d]$ generado por una distribución uniforme.

En SPSO 2011, la inicialización es casi la misma, sólo varía la asignación de velocidad:

$v_i(0) = U(\text{mín}_d - x_i(0), \text{máx}_d - x_i(0))$

2.3.3. Actualiza velocidad y posición

En SPSO 2006 y 2007, la velocidad se actualiza dimensión por dimensión de la siguiente manera:

$$v_{i,d}(t+1) = wv_{i,d}(t) + c_1 r_1 (p_i(t) - x_i(t)) + c_2 r_2 (l_i(t) - x_i(t)) \tag{2.4}$$

Los valores w, c_1 y c_2, son parámetros definidos por el usuario pero se recomienda utilizar:

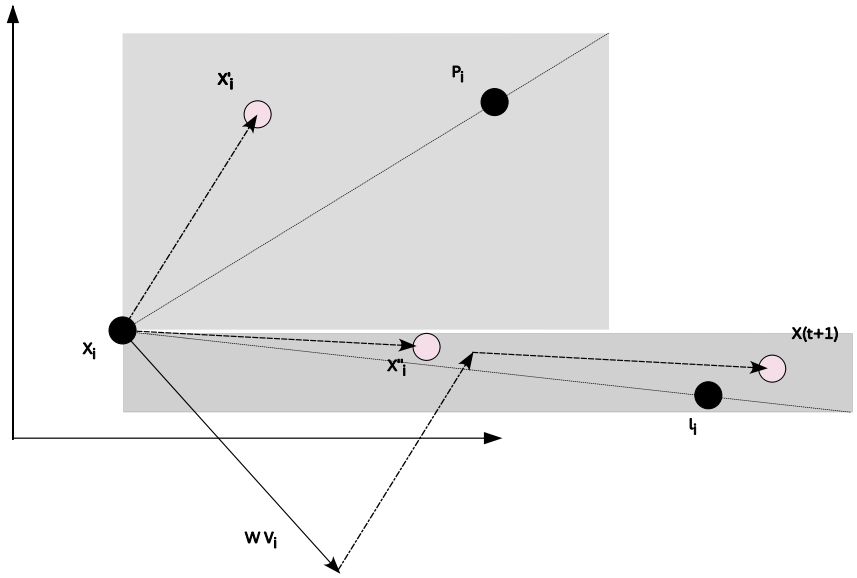

Figura 2.1: Construcción de la siguiente posición en SPSO 2006 y 2007. Los puntos x_i' y x_i'' son seleccionados de manera aleatoria dentro de dos paralelepípedos definidos por p_i y l_i con su distancia hacia x_i.

$c_1 = c_2 = \frac{1}{2} + \ln(2)$

$w = \frac{1}{2\ln(2)}$

Los valores r_1 y r_2 oscilan con distribución uniforme en el intervalo $[0, 1]$. Dada la velocidad, la siguiente posición se calcula con:

$$x_i(t+1) = x_i(t) + v_i(t+1) \tag{2.5}$$

La construcción de la siguiente posición de la partícula se ilustra en la figura 2.1.

En SPSO 2011:

Es en esta parte del algoritmo en la que el SPSO2011 difiere de manera significativa con respecto a sus antecesores. En esta versión se explota la idea del invariante rotacional (figura 2.2). Para cada partícula, en cada iteración, se define un centro de gravedad (G_i) el cual se construye con base en tres puntos: la posición actual de la partícula $(x_i(t))$, un punto (P_i) ligeramente alejado del $(p_i(t))$ y otro (L_i) ligeramente alejado del $(l_i(t))$, de la siguiente manera:

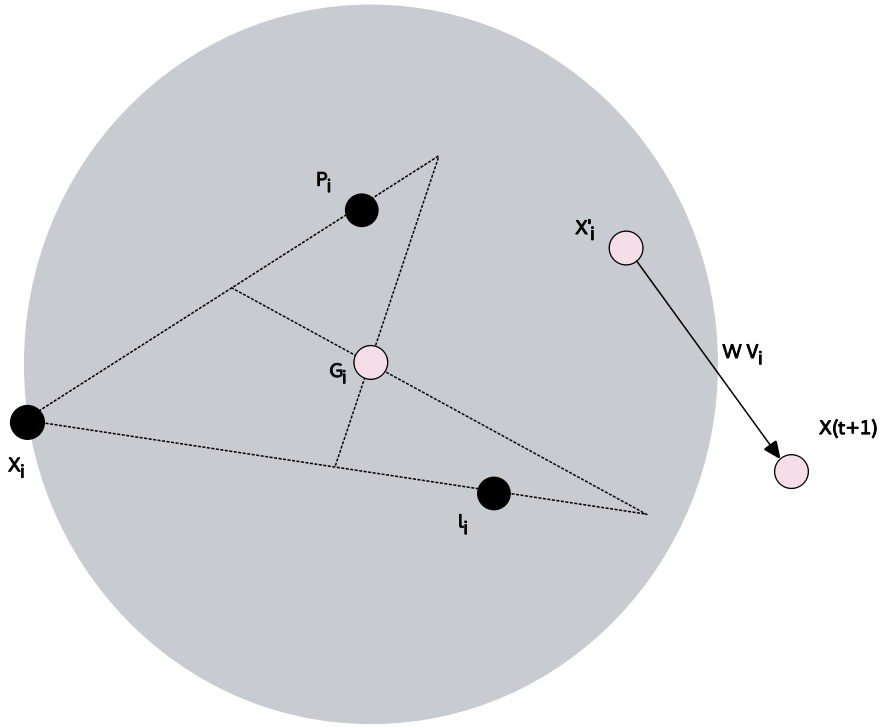

Figura 2.2: Construcción de la siguiente posición en SPSO2011. El punto x_i es seleccionado de manera aleatoria dentro de una hiper-esfera $H_i(G_i, \|G_i - x_i\|)$.

$$G_i = \frac{x_i + (x_i + c_1 r_1 (P_i - x_i)) + (x_i + c_2 r_2 (L_i - x_i))}{3} \tag{2.6}$$

Después se define un punto $x_i^{'}$ dentro de una hiper-esfera de centro G_i y radio $\|G_i - x_i\|$.

$$x_i^{'} = H_i(G_i, \|G_i - x_i\|) \tag{2.7}$$

Entonces, la velocidad y la nueva posición se actualizan usando:

$$v_i(t+1) = wv_i(t) + x_i^{'} - x_i(t) \tag{2.8}$$

$$x_i(t+1) = x_i(t) + v_i(t+1) \tag{2.9}$$

En el algoritmo 4 se muestra el pseudocódigo del SPSO2011.

Algoritmo 4: Pseudocódigo del SPSO2011

```
1  inicio
2      Inicialización del cúmulo
3      Localizar líder
4      g = 0
5      mientras g < Gmax hacer
6          para cada Partícula hacer
7              Definir centro de gravedad e hiper-esfera con las ecuaciones (2.6) y (2.7)
8              Actualizar velocidad y posición conforme a las ecuaciones (2.8) y (2.9)
9              Evaluar partícula
10             Actualizar pbest
11         fin
12         Actualizar líder
13         g++
14     fin
15 fin
```

2.3.4. Topologías de conexión entre partículas

Cada partícula selecciona a su líder conforme la información proporcionada por todo su vecindario, esto es, del vecindario se selecciona como líder a la mejor partícula. Definir el vecindario consiste en asignar una topología de conexión para conectar las partículas entre sí, y ésta determina la manera en la que se comparte información entre partículas. A continuación se presentan algunas de las topologías más utilizadas en PSO:

- **Anillo:** En esta topología cada partícula comparte su información personal con sus dos vecinos adyacentes, tál y como se muestra en la figura 2.3.

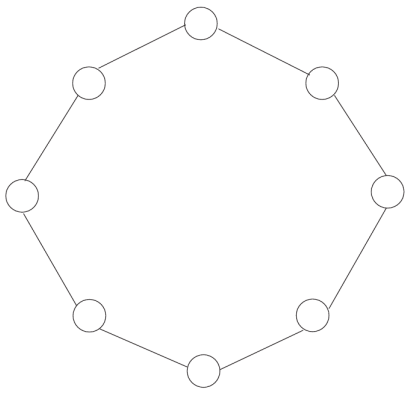

Figura 2.3: Topología de anillo, con un total de 8 partículas.

- **Estrella:** En esta topología, todas las partículas comparten su información personal

a una sola partícula llamada partícula focal. Un ejemplo de ella se muestra en la figura 2.4.

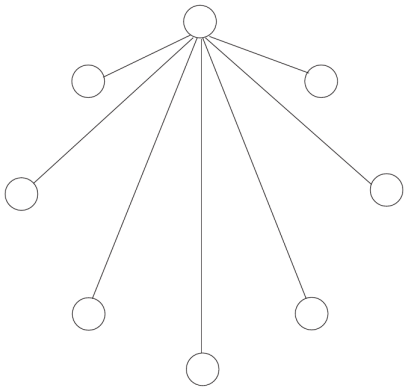

Figura 2.4: Topología de estrella, con un total de 8 partículas.

- **Totalmente conectada:** En esta topología todas las partículas en el cúmulo están conectadas unas con otras, tal y como se muestra en la figura 2.5.

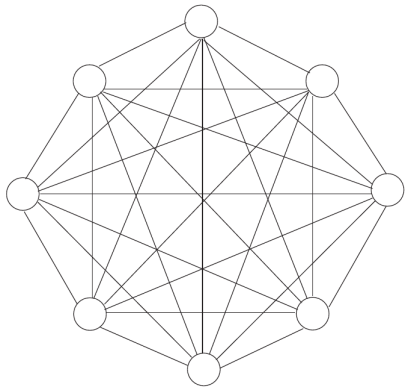

Figura 2.5: Topología totalmente conectada, con un total de 8 partículas.

- **Árbol:** En esta topología, cada partícula comparte su información personal con la partícula que se encuentra conectada a ella en un nivel superior y a un nivel inferior. en la figura 2.6 se muestra una topología de árbol con una altura igual a tres

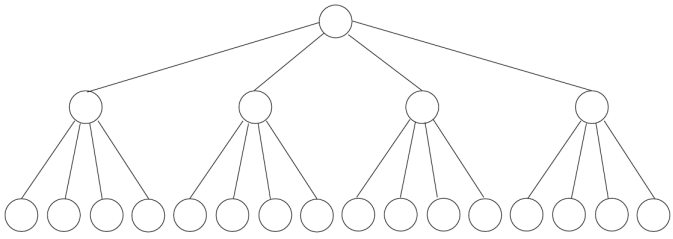

Figura 2.6: Topología de árbol, altura igual a 3, grado 4 y un total de 21 partículas.

2.3.5. Pseudocódigo general de PSO y MOPSO

En el algoritmo de PSO, una partícula es un vector que describe a una solución candidata. El algoritmo es iterativo, y en cada iteración cada partícula se "mueve" a través del paisaje de aptitud, haciendo uso de información personal (de sus partículas cercanas) y también de todo el cúmulo. El algoritmo general de un PSO de un solo objetivo es el que se muestra en el algoritmo 5.

Algoritmo 5: Pseudocódigo del PSO general

```
1  inicio
2      Inicialización del cúmulo
3      Localizar líder
4      g = 0
5      mientras g < Gmax hacer
6          para cada Partícula hacer
7              Actualizar velocidad y posición conforme a las fórmulas (2.4) y (2.5)
8              Evaluar partícula
9              Actualizar pbest
10         fin
11         Actualizar líder
12         g++
13     fin
14 fin
```

Para extender el PSO a problemas multi-objetivo es necesario hacer algunas modificaciones al algoritmo general. En este caso, el líder debe ayudar a la partícula a dirigirse hacia una posición no dominada. Para lograr esto, se mantiene un archivo de soluciones no dominadas, y es de este conjunto del cual se seleccionan los líderes para cada partícula. El algoritmo 6 muestra el pseudocódigo general de un *multi-objective particle swarm optimizer* (MOPSO).

Se hace notar que es práctica común dotar a un MOPSO de un operador de mutación (o turbulencia), el cual no existe en la versión mono-objetivo del PSO. Esto se debe a que en la versión multi-objetivo es muy importante mantener la diversidad de la población. Adicionalmente, los MOPSOs requieren de un mecanismo específico para mantener diversidad, el cual permita generar varias soluciones no dominadas diferentes en una sola ejecución algorítmica. Muchos MOPSOs utilizan un archivo externo (llamado también población secundaria) para almacenar las soluciones no dominadas generadas a lo largo del proceso de búsqueda. Este archivo cumple la función de mantener soluciones no dominadas que el re-cálculo de la velocidad del PSO podría perder (a esto se le llama elitismo). Adicionalmente, se permite el ingreso de soluciones al archivo externo solo cuando son no dominadas con respecto a su contenido o cuando dominan a soluciones del mismo (en ese caso, se borran del archivo las soluciones dominadas). Algunos MOPSOs usan un estimador de densidad en el archivo externo.

Algoritmo 6: Pseudocódigo del MOPSO general

```
1  inicio
2      Inicialización del cúmulo
3      Localizar líderes en un archivo externo
4      g = 0
5      mientras g < Gmax hacer
6          para cada Partícula hacer
7              Seleccionar el líder con base en la optimalidad de Pareto o un criterio
               análogo
8              Actualizar velocidad y posición
9              Mutar
10             Evaluar partícula
11             Actualizar pbest
12         fin
13         Actualizar los líderes del archivo externo
14         g++
15     fin
16     Reportar los resultados en un archivo externo
17 fin
```

2.3.6. Esquemas de selección en MOPSOs

Se han propuesto MOPSOs con diferentes esquemas de selección, de entre los cuales, los principales son los siguientes:

- **Basados en jerarquización de Pareto:** Seleccionan a sus líderes con base en la optimalidad de Pareto. Tal es el caso de SMPSO propuesto por Nebro y Durillo [3] en 2009, ellos proponen utilizar un coeficiente de constricción para evitar un sesgo de la velocidad de las partículas hacia el límite de las variables. La propuesta es muy buena en POMs con dos y tres objetivos pero por utilizar jerarquización de Pareto no escala adecuadamente a problemas con muchos objetivos.

- **Basados en indicadores de desempeño:** Adoptan un indicador de desempeño (normalmente) el hipervolumen en su mecanismo de selección de líderes o para mantener diversidad. Tal es el caso de MOPSOhv propuesto por García y Coello [19] en 2014. Este MOPSO utiliza la contribución al hipervolumen y con base en ello selecciona a las mejores partículas, teniendo un buen desempeño al resolver problemas de más de tres objetivos. R2-MOPSO propuesto por Fei Li [20] en 2015, utiliza el indicador R2 para seleccionar a sus mejores partículas, el cual logra muy buenos resultados al resolver problemas de dos o tres objetivos.

- **Basados en descomposición:** Transforman el problema multi-objetivo en varios problemas mono-objetivo que se resuelven simultáneamente. Tal es el caso del MOP-

SO/D propuesto por [21] en 2008, el cual implementa descomposición utilizando el método de Tchebycheff de manera muy similar a MOEA/D con la diferencia de que utiliza como motor de búsqueda el algoritmo de un MOPSO. Otro ejemplo es el dMOPSO propuesto por Zapotecas [4] en el 2011. Esta propuesta utiliza descomposición con el método denominado *Penalty Boundary Intersection* (PBI) [2] y un mecanismo de reinicialización de partículas.

Capítulo 3

Optimización evolutiva con muchos objetivos

Los algoritmos evolutivos multiobjetivos (AEMOs) han sido ampliamente utilizados para resolver problemas en la vida real. Sin embargo, los AEMOs basados en dominancia de Pareto tienen serias dificultades resolviendo problemas de cuatro o más objetivos a pesar de que tienen buen desempeño en dos y tres objetivos [14, 22]. Esto se debe principalmente a que conforme se incrementa la cantidad de objetivos en un problema de optimización multiobjetivo (POM), el espacio objetivo crece exponencialmente provocando que los individuos de la población en su mayoría sean soluciones no dominadas. Esto conlleva a la pérdida de presión de selección durante el proceso evolutivo, dificultando así la convergencia del conjunto de soluciones al frente de Pareto. Esta dificultad ha sido ampliamente analizada de manera analítica y experimental en la literatura especializada [23, 24].

Un problema de optimización con muchos objetivos (MaOPs por sus siglas en inglés) se define como un POM de cuatro o más objetivos. La dificultad que tienen los AEMOs para resolver MaOPs ha llamado el interés de una gran cantidad de investigadores dando lugar a una nueva generación de algoritmos evolutivos llamados *many-objective evolutionary algorithms* (MaOEAs). Las principales propuestas se han enfocado en el desarrollo de mecanismos de selección alternativos, la reducción del número de objetivos y la exploración de sólo algunas regiones del espacio de soluciones.

En la primera sección de este capítulo se explicarán los principales desafíos que enfrentan los MaOEAs al resolver problemas de más de cuatro objetivos. En la segunda y última sección se exponen los enfoques más representativos de las propuestas que hay hasta el momento para la solución de este tipo de problemas, haciendo énfasis en sus ventajas y limitaciones.

3.1. Dificultades en el manejo de muchos objetivos

Cuando incrementamos el número de objetivos, nos enfrentamos con las siguientes dificultades [25]:

- **Deterioro de la capacidad de búsqueda.** Provocado por el fenómeno de resistencia a la dominancia [26, 27, 28]. Una gran cantidad de soluciones no dominadas provoca que las soluciones no se puedan comparar usando dominancia de Pareto.
- **Representación del frente de Pareto.** En los problemas de optimización continuos con m objetivos en conflicto, el frente de Pareto es una variedad de dimensión (m - 1) [29, 30]. Para describir tal frente es necesario aumentar el tamaño de población exponencialmente con respecto a m.
- **Visualización del frente de Pareto.** La visualización del conjunto de soluciones en el espacio objetivo necesita de técnicas especiales, tales como la proyección a un espacio de menor dimensión y coordenadas paralelas [31].

En 1978, Bentley et al. [32] determinaron que en un conjunto de N vectores m-dimensionales, generados de manera aleatoria, el número esperado de vectores no dominados es de orden $O((logN)^{m-1})$.

Además, Farina y Amato [33] proponen la ecuación (3.1) para determinar, en un espacio de dimensión m, la proporción e de vectores no comparables (según la dominancia de Pareto) con respecto a un vector dado. De acuerdo a la ecuación (3.1) , a medida que aumenta el número de objetivos, la proporción de soluciones no comparables con respecto a una solución determinada crece de manera exponencial.

$$e = \frac{2^m - 2}{2^m} \tag{3.1}$$

La ecuación (3.1) también representa la forma en que se reduce la proporción de soluciones que pueden dominar a una solución determinada. Debido a esta reducción, la probabilidad de generar descendientes que dominen a sus ancestros disminuye de manera exponencial. De esta manera, Farina y Amato explican cómo la dominancia de Pareto puede resultar inadecuada para discriminar entre soluciones en espacios de búsqueda con muchos objetivos.

Teytaud et al. [34] compararon el límite inferior de tiempo computacional de ciertos AEMOs con respecto al límite superior del tiempo requerido por una búsqueda aleatoria, analizando AEMOs que utilizan dominancia de Pareto para discriminar soluciones. El análisis en [34] mostró que para los problemas considerados, cuando el número de objetivos se incrementa, el peor tiempo computacional de la búsqueda aleatoria es muy similar al mejor tiempo de los AEMOs estudiados. Por lo tanto, esos AEMOs son a lo mucho equivalentes a una búsqueda aleatoria cuando el número de objetivos es grande.

Sin embargo, en años recientes, algunos autores han manifestado que aumentar el número de objetivos de un problema no necesariamente lo hace más difícil. Brockhoff

et al. [35] realizaron un análisis teórico del tiempo de ejecución de un AEMO en un problema específico y mostraron cómo, al aumentar el número de objetivos, el problema puede volverse más o menos complejo. En tanto, Schütze et al. [36] llevaron a cabo un estudio teórico sobre la influencia del número de objetivos en la convergencia de un AEMO, concluyendo que agregar más objetivos a un problema puede hacerlo más difícil, pero no en todos los casos y no necesariamente de forma significativa.

3.2. Algoritmos evolutivos para la optimización con muchos objetivos

A continuación se presenta una clasificación de los MaOEAs divididos en siete clases de acuerdo a un estudio reciente [25]: basados en dominancia relajada, basados en diversidad, basados en agregación, basados en indicadores, basados en conjuntos de referencia, basados en preferencias, y basados en reducción de dimensionalidad.

3.2.1. Basados en dominancia relajada

El enfoque basado en dominancia relajada intenta relajar la definición de dominancia e incrementar la presión de selección hacia el frente de Pareto. Sin embargo, el incrementar la presión de selección podría hacer más difícil el mantener la diversidad. Para discriminar entre soluciones y mejorar la presión de selección hacia el frente de Pareto, distintas variantes de la dominancia de Pareto han sido propuestas, las cuales se pueden separar en dos clases: dominancia basada en valores y dominancia basada en números.

Dominancia basada en valores

Estos métodos modifican la dominancia de Pareto cambiando los valores de los objetivos de las soluciones cuando se comparan. Para ello se define un nuevo vector objetivo que corresponde a $F(x)$ como $G(x) = (g_1(x), g_2(x), ..., g_m(x))^T$. Esta modificación permite ampliar el área dominada de las soluciones no dominadas de modo que algunas de ellas son más propensas a ser dominadas por otras.

Laumanns et al. [37] propusieron el concepto de dominancia$-\epsilon$. Tomando en cuenta dos soluciones $x, y \in \Omega$ y $\epsilon > 0$, se dice que x $\epsilon-$domina y, si y sólo si $\forall i \in \{1, ..., m\}$, $(1 - \epsilon)f_i(x) \leq f_i(y)$. En este caso, $g_i(x) = (1 - \epsilon)f_i(x) \leq f_i(x)$, $\forall i \in \{1, ..., m\}$. Basados en el uso de la dominancia-ϵ, Deb et al. [38] propusieron un algoritmo con selección de estado uniforme denominado ϵ-AEMO. En ϵ-AEMO, el espacio objetivo es dividido en hipercajas, y a cada una se le asigna más de una solución. Las soluciones de diferentes hipercajas son comparadas con base en la dominancia-ϵ. Derivando el teorema de Sine, Sato [39] propone un método para controlar el área de dominancia de cada solución, el

cual se define de la siguiente manera:

$$g_i(x) = \frac{r \cdot \sin(\omega_i + S_i \cdot \pi)}{\sin(\omega_i + S_i \cdot \pi)} \tag{3.2}$$

En donde r es la norma de $F(x), \omega_i$ es el ángulo entre $F(x)$ y $f_i(y)$, S_i es un parámetro definido por el usuario para controlar el grado de contracción o expansión, y $\phi_i = S_i \cdot \pi$. Ikeda [40] propuso otra variante de la dominancia de Pareto llamada dominancia-α. Cuando se comparan dos soluciones en un objetivo, se toman en cuenta los valores de todos los objetivos. Esto permite que una solución domine a otra si es ligeramente inferior a ella en un objetivo pero altamente superior en otro objetivo.

Dominancia basada en números

Los métodos basados en números comparan las soluciones contando el número de objetivos en donde una solución es mejor, igual, o peor que otra. Farina y Amato [41] proponen la dominancia-k. Para dos soluciones $x, y \in \Omega, n_b, n_e, n_w$ representan la cantidad de objetivos donde x es mejor que, igual, o peor que y, respectivamente. Se dice que x domina-k a y si y sólo si

$$\begin{cases} n_e < m \\ n_b \geq \frac{m - n_e}{k+1} \end{cases} \tag{3.3}$$

En donde $0 \leq k \leq 1$. Además, cuando $k = 0$, esta relación de preferencia corresponde a la dominancia de Pareto. La relación de *favour* [42] es más simple que la dominancia-k. Una solución es considerada superior que otra en términos de la relación *favour* si $n_b > n_e$. Zou propone la dominancia-L (LD) [43], la cual puede ser vista como una extensión de la relación *favour* [42]. En términos de la dominancia-L, se dice que x domina a y si y sólo si

$$n_b - n_w = L > 0 \wedge \|F(x)\|_p < \|F(y)\|_p \text{ (para cierta } p) \tag{3.4}$$

Una solución es L-óptima si no hay otra solución que la domine con dominancia-L. Zou [43] demuestra que la dominancia de Pareto implica la dominancia-L, pero la dominancia-L no implica la dominancia de Pareto. Por lo tanto, el conjunto de L-óptimos es un subconjunto del conjunto de óptimos de Pareto y a su vez un frente de L-óptimos es un subconjunto del frente de Pareto.

Una de las ventajas de la dominancia basada en números es que la normalización de objetivos se hace automáticamente en la comparación de elementos. Esto puede resultar en mejoras significativas en la velocidad y simplicidad de los AEMOs [42].

3.2.2. Basados en diversidad

El enfoque basado en diversidad intenta mejorar el rendimiento de MaOEAs reduciendo el efecto negativo del mantenimiento de diversidad.

Adra y Fleming [44] presentan un mecanismo para administrar la diversidad nombrado DM1. DM1 determina si activar o no la promoción de diversidad de acuerdo a la distribución actual de la población. La promoción de diversidad se desactiva cuando la población es excesivamente diversa. Usando un conjunto de funciones de prueba de 20 objetivos, NSGA-II/DM1 supera en la mayor parte de los casos a NSGA-II en términos de convergencia y distribución.

Recientemente, se propuso una estrategia llamada estimación de densidad por desplazamiento (SDE por sus siglas en inglés) [45]. En específico, cuando se mide la densidad de un vecindario de soluciones, SDE desplaza la posición de una solución comparando su convergencia con la de las demás soluciones. Así, se toman en cuenta tanto la distribución como la información de la convergencia de las soluciones. Como resultado, esto puede reducir el impacto negativo de las soluciones resistentes a la dominancia en un AEMO basado en dominancia de Pareto.

3.2.3. Basados en agregación

Estos métodos se dividen en dos clases: métodos basados en agregación de información individual, y métodos basados en agregación de las comparaciones por pares.

Métodos basados en agregación de información individual

Esta clase de métodos normalmente utilizan la agregación de información individual para comparar soluciones. Empleando una serie de vectores de pesos, se puede descomponer un MaOP en muchos subproblemas de un objetivo, en donde cada uno corresponde a un vector de peso [46]. Como no utilizan dominancia de Pareto, pueden resolver MaOPs. Hay tres preguntas críticas con estos algoritmos: ¿Qué función de agregación se debe utilizar?, ¿cómo definimos los vectores de peso?, ¿cómo actualizo la mejor solución hasta el momento para cada subproblema?.

Zhang y Li [2] propusieron un algoritmo evolutivo basado en descomposición (MOEA/D). Este AEMO transforma el POM original en varios problemas mono-objetivo (usando un esquema de escalarización que se basa en el uso de pesos uniformemente distribuidos). Este algoritmo empareja a cada una de las soluciones con un único vector de pesos. Cada subproblema tiene un vecindario y es a través de él que se comparte información local para encontrar las mejores soluciones de cada subproblema. Este esquema tiene dos beneficios: baja complejidad computacional y un mecanismo que mejora sus resultados con el uso de la información de vecindarios locales. El cómo establecer los vectores de peso es una cuestión clave ya que éste determina la dirección de búsqueda. Hasta el momento, la selección de los vectores de peso sigue siendo un tema de investigación, aunque existen varios métodos en la literatura especializada [47].

El desempeño de estos algoritmos depende en gran medida del conjunto de vectores de peso utilizado y, sobre todo, de la función de agregación que se desee implementar.

Métodos basados en agregación de las comparaciones por pares

En estos métodos, la aptitud de una solución depende de las demás soluciones. Para una solución x, su valor de aptitud es la agregación de la comparación de x y todas las soluciones restantes en la población.

Una descripción general de estos métodos es la siguiente: $APC(x) = agg(comp(x, y))$, en donde $agg()$ es una función de agregación. El deterioro global (*global detriment* (GD) [48]), $comp(x, y)$ se define como $\sum_{k=1}^{m} max(f_k(x) - f_k(y), 0)$. Un mecanismo similar es utilizado en *Profit* (PF) [48], excepto que la calidad de la solución es expresada de acuerdo a su beneficio en lugar del detrimento. Los resultados en los problemas DTLZ con más de 50 objetivos mostraron que estos enfoques se comportaron mejor que seis métodos de asignación de aptitud del estado del arte [49] en términos de calidad de convergencia.

Para la función de aptitud *maxmin* [50], agg es la función de maximizacion, si $comp(x, y)$ es definida como $min_k(f_k(x) - f_k(y), 0)$. Entonces, una solución es dominada (no dominada, o débilmente dominada) si su valor de aptitud *maximin* es mayor que (menor que, igual a) cero. Menchaca y Coello [51] analizaron las propiedades de la función *maximin*, y propusieron el algoritmo *maximin-clustering multiobjective evolutionary algorithm* (MC-AEMO), que abarca tres operadores de selección, con el propósito de superar las desventajas de la función *maximin*. Los resultados experimentales mostraron que MC-AEMO es una buena elección para resolver POMs de baja y alta dimensionalidad.

3.2.4. Basados en indicadores

El enfoque basado en indicadores optimiza MaOPs guiándose por el valor de un indicador de desempeño sobre un conjunto de soluciones, de tal forma que un mejor valor del indicador significa que el conjunto de soluciones constituye una mejor aproximación del frente de Pareto verdadero. Estos métodos se pueden dividir en tres clases: basados en hipervolumen, basados en indicadores de distancia, y basados en el indicador R2.

Algoritmos basados en hipervolumen

El indicador de hipervolumen es de los que más han atraído la atención debido a su inherente consistencia con la definición de dominancia de Pareto [52]. Emmerich et al. [53] propusieron el *S-metric selection evlutionary multiobjetive algorithm* (SMS-EMOA) en el cual se reemplaza el operador de agrupamiento, (*crowding*) del NSGA-II por el hipervolumen, además de adoptarse una selección de estado uniforme. Éste es un algoritmo guiado por el gradiente del indicador de hipervolumen [54], en donde cada solución $a \in A$ es evaluada por su contribución al hipervolumen: $Con_{HV}(a, A) = I_{HV} - I_{HV}(A \setminus \{a\})$. Este algoritmo obtiene muy buenos resultados en baja dimensionalidad. Su principal desventaja es que su costo computacional se incrementa exponencialmente con el aumento de la dimensión del espacio objetivo. Tal como describe [55], el tiempo de ejecución del SMS-EMOA es $O(\mu^{\frac{m}{2}+1})$, en donde μ es el tamaño de población y m es el número de objetivos.

Bader y Zitzler [52] propusieron el *hypervolumen estimation algorithm* (HyPE) para lidiar con el alto costo computacional del hipervolumen. En este caso, se calcula una aproximación del hipervolumen utilizando el método de Monte Carlo, permitiendo así un buen compromiso entre precisión y tiempo de cómputo. Resultados experimentales muestran que HyPE alcanza un rendimiento competitivo en términos de hipervolumen en los problemas de prueba de los conjuntos *Deb-Thiele-Laumanns-Zitzler* (DTLZ), *Walking-Fish-Group* (WFG) y la mochila multi-objetivo (MKP, por sus siglas en inglés) para más de 50 objetivos. Sin embargo, otros investigadores han reportado un desempeño pobre de HyPE en problemas con muchos objetivos, lo que indica que su método de muestreo no es muy efectivo (ver por ejemplo [56]). Calcular el valor exacto de este indicador implica un muy alto costo computacional, por lo cual algunos investigadores prefieren utilizar otros indicadores.

Algoritmos basados en indicadores de distancia

Estudios experimentales de estos algoritmos han demostrado que son bastante competitivos. Bringmann [57] propone el *approximation-guided evolutionary algorithm* (AGE) que tiene como objetivo minimizar la evaluación del indicador-α. Otro algoritmo es el propuesto por Denysiuk [58] llamado *many-objective differential evolution with mutation restriction* (MyO-DEMR) el cual es muy bueno en la evaluación de IGD. El rendimiento de estos algoritmos ha demostrado ser bastante competitivo. Por ejemplo, AGE presenta mejores resultados arriba de 20 objetivos en términos del indicador-α y del hipervolumen en comparación con IBEA [59], NSGA-II [60], SMS-EMOA [61], y SPEA2 [62].

Algoritmos basados en el indicador R2

Manriquez propuso R2-MOGA y R2MODE [63] que integran el indicador R2 en una versión modificada del método de ordenamiento no dominado de Goldberg [64]. Evaluaciones de estos algoritmos en los problemas DTLZ muestran que para más de 10 objetivos presentan mejores resultados que SMS-EMOA, requiriendo un tiempo mucho menor. Otro algoritmo basado en el indicador R2 es el presentado por Hernández y Coello [65] llamado *many-objective metaheuristic based on R2 indicator* (MOMBI) el cual alcanza un rendimiento muy similar a los antes mencionados. Posteriormente, Hernández y Coello [66] proponen una segunda versión de MOMBI, llamado MOMBI-II el cual logra mejorar la diversidad de las soluciones en problemas de muchos objetivos, en comparación con su versión anterior. Además, Phan y Suzuki [67] proponen R2-IBEA, el cual está diseñado para obtener una buena aproximación al frente de Pareto mediante la corrección de un sesgo inherente en el indicador R2. El uso de el indicador R2 ha sido recomendado para resolver problemas de más de cuatro objetivos [68], ya que evalúa de manera simultanea la convergencia y la diversidad del conjunto de soluciones.

3.2.5. Basados en conjuntos de referencia

Este enfoque usa un conjunto de referencia para estimar la calidad de una solución y así guiar el proceso de búsqueda. En este enfoque hay dos puntos clave: definir el conjunto de referencia y el cómo utilizarlo para estimar la calidad de la población. Estos algoritmos los podemos clasificar en dos tipos: los que utilizan un conjunto de referencia reales, y los que utilizan un conjunto de referencia virtuales.

Enfoque basado en conjunto de referencia reales

Praditwong y Yao [69] propusieron un algoritmo basado en el uso de dos archivos (TAA), el cual separa las soluciones no dominadas de cada generación en dos archivos, llamados archivo de convergencia (AC) y archivo de diversidad (AD). En ellos se almacenan las soluciones para promover convergencia y diversidad, respectivamente. Cuando las soluciones totales de la unión de AC y AD superan un cierto límite pre-establecido por el usuario, las soluciones en AD con menor distancia hacia AC son removidas hasta alcanzar el límite de soluciones. Estudios experimentales [70] mostraron que TAA supera a otros MaOEAs del estado del arte de manera significativa en convergencia y muestra ser competitivo en diversidad. La última variante de este algoritmo es (Two_Arch2) propuesto por Wang [70].

Enfoque basado en un conjunto de referencia virtual

A diferencia de TAA, el NSGA-II para muchos objetivos (NSGA-III) [71] y *taxi-cab surface evolutionary algorithm* (TC-SEA) [72] construyen un conjunto de referencia de manera similar, pero se diferencian en sus mecanismos de selección. Para escalar el NSGA-II a MaOPs, se reemplaza la distancia de agrupamiento (*crowding*) que es un operador para preservar la diversidad, por una estrategia de preservación de nichos basada en un conjunto de referencia [71]. NSGA-III tiene un buen desempeño en problemas de más de 10 objetivos [71]. El TC-SEA [72] difiere respecto a NSGA-III en tres aspectos. Primero, TC-SEA construye el conjunto de referencia durante la ejecución, mientras que NSGA-III usa un conjunto de referencia predefinido. Segundo, TC-SEA asocia soluciones al conjunto de referencia de acuerdo a su valor TC en lugar de la distancia ortogonal como en NSGA-III. Tercero, TC-SEA usa la clasificación de los valores TC como segundo criterio de selección.

3.2.6. Basados en preferencias

Para aproximar el frente de Pareto usando MaOEAs, necesitamos incrementar el tamaño de población de manera exponencial en términos de la dimensionalidad del espacio objetivo [73]. Sin embargo, en la mayoría de las aplicaciones de MaOPs del mundo real, el tamaño de la población está muy limitado en comparación con la cantidad de soluciones necesarias para un resultado significativo [74]. Por ello, se considera una buena idea el

enfocarse en un subconjunto del frente de Pareto de acuerdo a las preferencias del usuario. Este tipo de enfoques ya ha sido discutido en la literatura especializada [73].

Hay dos puntos claves en los enfoques basados en preferencias: qué modelo de preferencia utilizar y cuándo integrar la información sobre las preferencia del usuario. Existe una gran variedad de modelos de preferencias, tales como especificación de metas, compromiso entre objetivos, objetivos con preferencias, entre otros [74]. De acuerdo al momento en el que se integra la información de las preferencias en el proceso de optimización, se pueden definir tres clases de algoritmos [75, 76]:

- **Algoritmos a priori (antes de la búsqueda)**: La información sobre las preferencias se configura antes de la búsqueda, y éstas guían la convergencia de la población hacia la región de interés del frente de Pareto [77, 78, 79, 80].
- **Algoritmos interactivos (durante la búsqueda)**: Durante el proceso de optimización se le pide al tomador de decisiones la información de las preferencias de manera interactiva para dirigir la búsqueda hacia una cierta región del frente de Pareto [81, 82, 83, 84, 76, 85]
- **Algoritmos a posteriori (después de la búsqueda)**: Las preferencias son introducidas al final del proceso de optimización y es hasta entonces que se aplican al conjunto que aproxima al frente de Pareto [86, 87, 88, 89].

3.2.7. Basados en reducción de dimensionalidad

El enfoque de reducción de la dimensionalidad trata de reducir el número de objetivos y cambiar el MaOP a uno con menos objetivos. Este tipo de enfoques se pueden clasificar de acuerdo al momento en el que se incorpora una técnica de reducción de dimensionalidad en el proceso de optimización en dos grandes grupos: en línea y fuera de línea.

Reducción de dimensionalidad fuera de línea

En los métodos fuera de línea, la reducción de dimensinalidad toma acción al obtener un conjunto de óptimos de Pareto. De acuerdo a la técnica de reducción de dimensionalidad utilizada se pueden clasificar estos algoritmos en tres clases:

- **Métodos basados en correlación**: Esta técnica consiste en examinar la correlación entre los objetivos. Saxena [90] propuso L-PCA basado en análisis de componentes principales para reducción lineal y NL-MVU-PCA basado en despliegue de máxima varianza para reducción no lineal.
- **Métodos basados en estructuras de dominancia**: Estos métodos permiten reducir el número de objetivos considerando la relación de dominancia entre las soluciones obtenidas en una ejecución de un MaOEA. Brockhoff y Zitzler [91] propusieron una heurística que de manera iterativa combina dos objetivos usando un método de combinación ponderada. El método demostró ser altamente útil para reducir la pérdida de información.

- **Métodos basados en selección de características**: Jaimes y Coello [92] propusieron dos algoritmos para reducir el número de objetivos basándose en una técnica de selección de características de Mitra [93] Ambos métodos seleccionan objetivos esenciales conforme la correlación entre cada par de objetivos de modo que se descartan menos objetivos en conflicto.

Reducción de dimensionalidad en línea

Con la obtención de conjuntos de soluciones de forma iterativa e invocando la técnica de reducción de dimensionalidad, el número de objetivos puede ser reducido gradualmente durante el proceso de búsqueda. Hasta el momento [94], todos los frentes de Pareto de algoritmos con reducción de dimensionalidad en línea siguen un patrón basado en correlación. C-PCA-NSGA-II [95], MVU-PCA-NSGA-II [95] y PCA-NSGA-II [96] usan la misma base para obtener conjuntos de soluciones de manera iterativa y reducir los objetivos utilizando información de correlación [95, 96].

En general, los métodos de reducción de dimensionalidad pueden reducir de manera significativa el costo computacional pero pierden algo de información a causa de la reducción del número de objetivos.

Como pudo verse en este capítulo, existe una amplia gama de técnicas con las cuales un AEMO puede resolver adecuadamente problemas con muchas funciones objetivo. Evidentemente, cada uno de estos enfoques presentan ciertas ventajas, pero a la vez, conllevan ciertas desventajas. Buscando contar con una técnica simple de implementar y en la que resultara fácil acoplar un optimizador mono-objetivo, decidimos adoptar un método de descomposición para fines del algoritmo que se propone en esta tesis, tal y como se detalla en el capítulo siguiente.

Capítulo 4

Algoritmo propuesto

En este capítulo se describe el MOPSO propuesto como parte del trabajo de tesis, el cual se puede usar para resolver problemas de optimización multi-objetivo (POMs) continuos y sin restricciones con pocas o muchas funciones objetivo. Dicho algoritmo está basado en descomposición y su característica más representativa es que utiliza la fórmula de velocidad del SPSO2011, la cual le atribuye al algoritmo la propiedad de ser invariante a la rotación. El algoritmo que presentamos, se denomina *decomposition-based multi-objective particle swarm optimization algorithm based on SPSO2011* (DMOPSO11).

Este capítulo se divide en 5 secciones en las que se describe a detalle el funcionamiento del DMOPSO11, a partir del algoritmo 7. En la primera sección se muestra la notación necesaria para la explicación del algoritmo. La sección 2 explica todos los aspectos relacionados con la inicialización del algoritmo. En la sección 3 se explica la actualización de las mejores partículas globales. La sección 4 describe el proceso para actualizar la posición de cada partícula. Por último, la sección 5 explica cómo se actualiza el mejor personal.

4.1. Notación necesaria

En esta sección se presenta la notación necesaria y el pseudocódigo del DMOPSO11, esto con el fin de dar un correcto seguimiento a la explicación del DMOPSO11 a lo largo de este capítulo. La notación es la siguiente:

- t : Generación actual dentro del algoritmo.
- N : Tamaño de la población.
- $\vec{W}$: Vector que almacena los N vectores de peso.
- v_i : i-ésimo vector de peso dentro de $\vec{W}$, donde $i = 1, .., N$.
- P^t : Cúmulo de partículas, de tamaño N, en la generación t
- x_i : Posición de la i-ésima partícula dentro del cúmulo P^t.

- v_i : Velocidad de la i-ésima partícula.
- a_i : Edad de la i-ésima partícula.
- $x_{pb,i}$: Mejor posición personal de la i-ésima partícula.
- $x_{lb,i}$: Mejor posición del líder de la i-ésima partícula.
- $x_{i,obj}$: Evaluación de la i-ésima partícula.
- $Gbest$: Conjunto que almacena a los posibles líderes.
- T : Tamaño del vecindario.
- B_i : Vecindario de la partícula i.
- T_a : Edad máxima permitida para una partícula.
- z^* : Vector ideal.
- z^{nad} : Vector nadir.
- k : Número de objetivos del POM.
- $superior_j$: Valor superior que puede alcanzar la variable j.
- $inferior_j$: Valor inferior que puede alcanzar la variable j.
- $numVars$: Número de variables del POM.

Algoritmo 7: Pseudocódigo del DMOPSO11

inicio
  $t = 0$
  Generar conjunto de N vectores de peso $\vec{W} = \{w_1, ...w_N\}$
  Generar los vecindarios B_i (donde $i = 1, .., N$)
  Generar cúmulo $P^t = \{x_1, ..., x_N\}$ de N partículas
  Inicializar la velocidad v_i y la edad a_i para cada partícula x_i (donde $i = 1, .., N$)
  Evaluar las N partículas e iniciamos los vectores z^*, z^{nad}
  Definir el mejor personal: $x_{pb,i} = x_i$, $Gbest = \emptyset$
  mientras $t < Gmax$ **hacer**
    Actualizar *Gbest*
    para cada *Partícula* x_i **hacer**
      Actualizar el líder $x_{lb,i}$
      si $(a_i{}^t < T_a)$ **entonces**
        Actualizar y Ajustar $v_i{}^{t+1}$
        Calcular la nueva posición $x_i{}^{t+1}$
        Mutar $x_i{}^{t+1}$
      fin
      en otro caso
        Reiniciar la edad de la partícula $a_i{}^{t+1} = 0$
        Reiniciar la posición $x_i{}^{t+1}$
      fin
      Evaluar la partícula, obtener $F(x_i{}^{t+1})$
      si $(g(x_i|w_i, z^*, z^{nad}) < g(x_{pb,i}|w_i, z^*, z^{nad}))$ **entonces**
        Actualizar la mejor posición personal $x_{pb,i} = x_i$
        Reiniciar la edad de la partícula $a_i{}^{t+1} = 0$
      fin
      en otro caso
        Aumentar la edad de la partícula $a_i{}^{t+1} = a_i{}^t + 1$
      fin
      si $a_i{}^t \neq 0$ **entonces**
        para cada *Partícula* $x_j \in B_i$ **hacer**
          si $(g(x_i|w_j, z^*, z^{nad}) < g(x_{pb,j}|w_j, z^*, z^{nad}))$ **entonces**
            Actualizar la mejor posición personal de la partícula vecina $x_{pb,j} = x_i$
            Reiniciar la edad de la partícula vecina $a_j{}^{t+1} = 0$
          fin
        fin
      fin
      Actualizamos z^* y z^{nad}
    fin
    $t \leftarrow t + 1$
  fin
  Reporta las partículas almacenadas en *Gbest*
fin

4.2. Iniciando el DMOPSO11

Nuestro algoritmo utiliza el enfoque de descomposición. Esto significa que el problema de optimización multi-objetivo se transforma en un conjunto de subproblemas de optimización escalar, de tal manera que la solución de cada subproblema represente una región distinta del frente de Pareto. Para llevar esto a cabo, se define una función de escalarización. En nuestro caso, adoptamos la *achievement scalarizing function* (ASF) propuesta por Miettinen [97] debido a que ha demostrado tener varias ventajas en comparación con otras técnicas de escalarización como se indica en [97]. Es por esta razón que implementamos ASF en DMOPSO11.

Dentro del algoritmo, el $Gbest_i$ corresponde a la solución del i-ésimo subproblema, siendo $Gbest$ el contenedor del conjunto de soluciones que avanzan hacia el conjunto óptimo de Pareto durante la minimización. Dicho lo anterior, podemos explicar los pasos para la inicialización del algoritmo:

- Inicializar los vectores de peso:
  Se define un conjunto W de tamaño N de vectores de peso de dimensión k bien distribuidos utilizando el método de *simplex-lattice*.

- Inicializa los vecindarios B_i:
  B_i almacena los T índices de los vecinos de la partícula i ($B_i = B_{i,j}, ..., B_{i,T}$). Para asignar el índice j al vecindario B_i el vector de peso w_j debe ser parte de los T vecinos mas cercanos al vector de peso w_i.

- Inicializar la población:
  El conjunto de N partículas $P^t = \{x_1, ..., x_N\}$ se inicializa de manera aleatoria con una distribución uniforme y se establece una velocidad de $v_{i,j} = 0$ para cada partícula i, considerando $j = \{1, ..., k\}$.

- Evalúa la población:
  Cada partícula en P^t es evaluada usando: $x_{i,obj} = F(x_i) = \{f_1(x_i), ..., f_k(x_i)\}$.

- Iniciamos los vectores z^* y z^{nad}:
  z^* :
  Si $f_j({x_i}^t) < {z_j}^*$ entonces ${z_j}^* = f_j({x_i}^t)$ (para $j = 1, ..., k$)

  ${z}^{nad}$:
  Si $f_j({x_i}^t) > {z_j}^{nad}$ entonces ${z_j}^{nad} = f_j({x_i}^t)$ (para $j = 1, ..., k$)

- Define la mejor posición personal para cada partícula:
  $x_{pb,i} = x_i$.

- Define el conjunto de mejores posiciones globales $Gbest$:
  $Gbest = \emptyset$.

4.3. Actualizando la posición de cada partícula

Esta es la parte más significativa del DMOPSO11. En esta etapa el algoritmo implementa la fórmula de velocidad del SPSO2011 que es la que representa el motor de búsqueda. Se implementa el coeficiente de constricción propuesto en SMPSO [98] el cual permite evitar sesgos en la velocidad con el propósito de mejorar el desempeño en problemas multimodales. Además, se utiliza un mecanismo propuesto en dMOPSO [4] para reinicializar la partícula en caso de que ésta ya no esté obteniendo buenos resultados. A continuación se explica paso a paso el proceso para actualizar la posición de una partícula.

En caso de que la edad de la partícula no haya alcanzado el máximo, esto es que $a_i < T_a$, la velocidad y posición de una partícula se actualizan de la siguiente manera:

- Selección del líder:
  Como ya se ha mencionado, el líder se selecciona del conjunto *Gbest*. Esto se hace de dos maneras: 50 % de las veces se selecciona al azar entre todo el conjunto *Gbest*, y el otro 50 % se toma de igual forma al azar pero de los índices que representan los subproblemas vecinos. Lo anterior es con el propósito de balancear un poco más la diversidad en este proceso.

- Actualiza velocidad y posición:
  Primero se establecen los coeficientes C_1 y C_2 de manera aleatoria en el rango [1.5,2.5], esto es para que el coeficiente de constricción pueda ser calculado de la siguiente manera:

$$\chi = \frac{2}{2 - \varphi - \sqrt{\varphi^2 - 4\varphi}} \tag{4.1}$$

  en donde:

$$\varphi = \begin{cases} C_1 + C_2 & \text{si } C_1 + C_2 > 4 \\ 4 & \text{si } C_1 + C_2 \leq 4 \end{cases} \tag{4.2}$$

  Se define un centro de gravedad (G_i) el cual se construye con base en tres puntos: la posición actual de la partícula ($x_i(t)$), un punto (P_i) ligeramente alejado del ($p_i(t)$) y otro (L_i) ligeramente alejado del ($l_i(t)$), de la siguiente manera:

$$P_i = x_i(t) + C_1 U_1 \bigotimes (p_i(t) - x_i(t))$$
$$L_i = x_i(t) + C_2 U_2 \bigotimes (l_i(t) - x_i(t))$$

$$G_i = \frac{x_i(t) + P_i + L_i}{3}$$

Siendo U_1 y U_2 números en el rango [0.0,1.0]. Después se define un punto $x_i^{'}$ dentro de una hiper-esfera de centro G_i y radio $\|G_i - x_i\|$.

$x_i^{'} = H_i(G_i, \|G_i - x_i\|)$

Entonces, la velocidad se actualiza:

$v_i(t+1) = \chi(wv_i(t) + x_i^{'} - x_i(t))$

Después se ajusta en caso de que la velocidad se pase de los límites permitidos de la siguiente manera:

$$v_{i,j}(t) = \begin{cases} \delta_j & \text{si } v_{i,j}(t) > \delta_j \\ -\delta_j & \text{si } v_{i,j}(t) \leq -\delta_j \\ v_{i,j}(t) & \text{caso contrario} \end{cases} \tag{4.3}$$

en donde:

$$\delta_j = \frac{(superior_j - inferior_j)}{2} \tag{4.4}$$

Por último, la nueva posición se asigna de la siguiente manera:

$x_i(t+1) = x_i(t) + v_i(t+1)$

- Se aplica mutación: Cada partícula es mutada usando mutación polinomial [99], con probabilidad de $\frac{1}{numVars}$

En caso de que la edad de la velocidad de la partícula alcance el máximo, esto es que $a_i \geq T_a$, la posición se actualiza de la siguiente manera:

- Reinicia velocidad y edad:
  $v_{i,j}^{t+1} = 1$
  $a_{i,j}^{t+1} = 0$

  para cada $j = 1, ..., k$

- Reinicia posición:
  95 % de las veces se reinicia:

$$x_{i,j}^{t+1} = N(\frac{x_{lb,i,j} + x_{pb,i,j}}{2}, |x_{lb,i,j} - x_{pb,i,j}|) \tag{4.5}$$

  el 5 % restante se reinicia:

$$x_{i,j}^{t+1} = N(\frac{x_{lb,i,j} - x_{pb,i,j}}{2}, |x_{lb,i,j} - x_{pb,i,j}|) \tag{4.6}$$

4.4. Actualizando la mejor posición personal y el conjunto *Gbest*

Las líneas 26 y 27 del algoritmo 7 corresponden a la actualización de la mejor posición personal. En este paso se compara el desempeño de la nueva posición de la partícula contra el de la mejor posición personal que ha tenido hasta ese momento en el subproblema que le corresponde. En caso de que la nueva posición sea mejor que la anterior, la nueva posición pasa a tomar su lugar, es decir:

si

$$g(x_i|w_i, z^*, z^{nad}) < g(x_{pb,i}|w_i, z^*, z^{nad}).$$

para

$$g(x_i|w_i, z^*, z^{nad}) = max_m^{j=1} \left\{ \frac{1}{w_{i,j}} \left| \frac{f_j(x_i) - z_j^*}{z_j^{nad} - z_j^*} \right| \right\}$$
$$g(x_{pb,i}|w_i, z^*, z^{nad}) = max_m^{j=1} \left\{ \frac{1}{w_{i,j}} \left| \frac{f_j(x_{pb,i}) - z_j^*}{z_j^{nad} - z_j^*} \right| \right\}$$

entonces

$$x_{pb,i} = x_i$$

Esta información personal debe ser compartida con el fin de que pueda mejorar la búsqueda de otras partículas. Considerando que cada partícula busca solucionar un problema escalar en particular enfocándose en una región en particular, se estableció que cada partícula compartiese su mejor posición personal con sus T-vecinos más cercanos; así es más probable que le sea de utilidad a los vecinos. El problema que surge es que la diversidad se puede llegar a perder de manera prematura en ciertos casos en los que la mejor posición personal sea compartido por varias partículas debido a una posible resistencia a la dominancia. De acuerdo a esta conjetura, se optó por solo compartir su mejor posición si y solo si fracasó al mejorar su mejor posición personal durante la iteración

actual. Por lo tanto, en la mayoría de los casos está compartiendo posiciones cercanas a su mejor posición.

El conjunto *Gbest* es el que almacena las soluciones que minimizan cada subproblema y del cual también se seleccionan los líderes durante la optimización. *Gbest* se actualiza al inicio de la iteración, el algoritmo 8 describe la manera en la que se efectúa la actualización de *Gbest*.

Algoritmo 8: *ActualizaGbest*

inicio
  $Gbest = \emptyset$
  para cada $w_i \in W$ **hacer**
    $Gbest = Gbest \cup \{x_j | \min_{x_j \in P^t} g(x_j | w_i, z^*, z^{nad})\}$
    $P^t = P^t \setminus x_j$
  fin
  Regresa *Gbest*
fin

Capítulo 5

Estudio experimental

En este capítulo se presentan los resultados obtenidos en la evaluación del algoritmo propuesto, el cual decidimos llamar *decomposition-based multi-objective particle swarm optimization algorithm based on SPSO2011* (DMOPSO11). Se evaluó DMOPSO11 tomando en cuenta diferentes problemas de prueba que representan frentes de Pareto con diferentes características incluyendo frentes cóncavos, convexos, discontinuos, y multi-frontales. Entre las familias de problemas de prueba adoptadas están el conjunto *Zitzler-Deb-Thiele* (ZDT) que consiste de problemas de dos objetivos, sin incluir ZDT5 que es un problema con variables discretas. También adoptamos los conjuntos *Deb-Thiele-Laumanns-Zitzler* (DTLZ) y *Walking-Fish-Group* (WFG). Las tres familias de prueba utilizadas (ZDT, DTLZ y WFG) se encuentran descritas en el apéndice A . Por último, se midió la escalabilidad de dos a diez objetivos con DTLZ2. Para evaluar el desempeño de nuestra propuesta los resultados han sido medidos con los indicadores que se describen en el apéndice B, y comparamos los resultados con respecto a un AEMO del estado del arte *multiobjective evolutionary algorithm based on decomposition* (MOEA/D) [2]. También se compararon resultados con respecto a dos MOPSOs del estado del arte: *Speed-constrained Multi-objective PSO* (SMPSO) [3] y *decomposition-based multi-objective particle swarm optimizer* (dMOPSO) [4]. En la sección 5.1 se explica la metodología y configuración de los problemas de prueba de la experimentación. En la sección 5.2 se explican los parámetros utilizados en cada algoritmo. Por último, en la sección 5.3 se presentan los resultados obtenidos, el análisis de escalabilidad y la comparación de resultados con respecto a MOEA/D, SMPSO y dMOPSO.

5.1. Diseño experimental

El principal objetivo de este capítulo es evaluar el algoritmo propuesto (DMOPSO11) al resolver problemas de optimización multi-objetivo (POMs) para encontrar sus ventajas y desventajas en comparación con AEMOs y MOPSOs representativos del estado del arte. Para ello, se propuso comparar DMOPSO11 contra MOEA/D que es un algoritmo genético multiobjetivo basado en descomposición. Así también se compara contra dos

MOPSOs del estado del arte: SMPSO y dMOPSO. Se utilizaron 21 problemas de prueba correspondientes a los conjuntos ZDT, DTLZ y WFG. El conjunto ZDT se adoptó con dos objetivos, mientras que los conjuntos DTLZ y WFG se adoptaron con tres objetivos. Además, se presenta un análisis de escalabilidad utilizando DTLZ2 con 2 a 10 objetivos. Para lograr resultados más confiables, cada AEMO se ejecutó 30 veces para solucionar cada instancia del problema, reportándose la media y la desviación estándar de los indicadores I_{HV}, I_{IGD+} e I_S presentados en el apéndice B.

Tabla 5.1: Parámetros utilizados en los problemas de prueba

ZDT1-3	ZDT4, ZDT6	DTLZ1-6	DTLZ7	WFG1-9
$n = 30$	$n = 10$	$k = 10$ $n = m + k$	$k = 20$ $n = m + k$	$k = 2(m - 1)$ $l = 20$ $n = k + l$

Para evaluar el desempeño de un AEMO es necesario utilizar distintos indicadores de desempeño, los cuales nos permitan ver su capacidad de convergencia y distribución de las soluciones. Para medir el desempeño en convergencia se utilizaron dos indicadores: el hipervolumen (I_{HV}) y la distancia generacional invertida más (I_{IGD+}). Para medir la distribución se utilizó el indicador de espaciamiento (I_S). El número de variables (n) utilizado para cada POM se presenta en la tabla 5.1. En la tabla 5.2 se muestran los puntos de referencia en el espacio objetivo utilizados para calcular el hipervolumen de las soluciones obtenidas en cada problema. Para calcular la distancia generacional invertida más se utilizaron $10,000m$ muestras del frente de Pareto verdadero obtenidos del repositorio de EMOO [100].

Tabla 5.2: Puntos de referencia utilizados para calcular el hipervolumen en cada problema de prueba de m objetivos.

Problema de prueba	Puntos de referencia $r = [r_1, ..., r_m]$
ZDT1-6	$[1.1, 1.1]$
DTLZ1	$[1, ..., 1]$
DTLZ2,DTLZ4	$[2, ..., 2]$
DTLZ3	$[7, ..., 7]$
DTLZ5	$[4, ..., 4]$
DTLZ6	$[11, ..., 11]$
DTLZ7	$[1, ..., 1, 2m]$
WFG1-9	$[3, 5, 7, ..., 2m + 1]$

5.2. Parámetros utilizados

En esta sección se presentan los parámetros utilizados en la fase experimental, los cuales se muestran en la tabla 5.3. En el caso de MOEA/D [2], SMPSO [3] y dMOPSO [4] se utilizaron los valores recomendados por sus respectivos autores. DMOPSO11, SMPSO y dMOPSO utilizan los parámetros propios de un PSO como el factor inercial w, el coeficiente personal y social, c_1 y c_2 respectivamente. En el caso de DMOPSO11, también se utilizan algunos parámetros adicionales a los de un PSO, tales como el tamaño del vecindario T porque utiliza vecindarios para compartir la información entre partículas. Así mismo, se agrega el parámetro $maxAge$ ya que utiliza una fórmula para reiniciar partículas en caso de que éstas no presenten mejoría en un número determinado de generaciones. Por último, como DMOPSO11 utiliza mutación polinomial, se tienen que definir la probabilidad de mutación P_m y el índice de distribución η_m.

Tabla 5.3: Valores de los parámetros de cada AEMO

DMOPSO11	MOEA/D	SMPSO	dMOPSO
$w = 0.5$	$P_c = 1.0$	$w = U(1.0, 5.0)$	$w = U(1.0, 4.0)$
$c1 = U(1.5, 2.5)$	$P_m = 1/n$	$c1 = U(1.5, 2.5)$	$c1 = U(1.5, 2.5)$
$c2 = U(1.5, 2.5)$	$\eta_c = 20$	$c2 = U(1.5, 2.5)$	$c2 = U(1.5, 2.5)$
$r1 = U(0, 1)$	$\eta_m = 20$	$r1 = U(0, 1)$	$r1 = U(0, 1)$
$r2 = U(0, 1)$	$T = 20$	$r2 = U(0, 1)$	$r2 = U(0, 1)$
$T = 20$			$maxAge = 2$
$maxAge = 2$			
$P_m = 1/n$			
$\eta_m = 20$			

En la tabla 5.4 se muestra el tamaño de la población y el número de generaciones para cada número de objetivos que utiliza cada AEMO al resolver los problemas ZDT, DTLZ y WFG. El tamaño de la población utilizado para la experimentación no crece de manera significativa al aumentar el número de objetivos. Lo anterior se debe a que estos algoritmos basados en descomposición establecen el tamaño de población según la cantidad de vectores de peso generados, siendo estos últimos los que siguen un incremento no lineal en cuanto a la cantidad de objetivos.

Tabla 5.4: Tamaño de población y número de generaciones utilizados en la ejecución de los algoritmos para cada número de objetivos

Número de objetivos	Tamaño de la población	Número de generaciones	Total de evaluaciones de la función objetivo
2	100	220	22000
3	120	230	27600
4	120	240	28800
5	126	250	31500
6	126	260	32760
7	210	270	56700
8	120	280	33600
9	165	290	47850
10	220	300	66000

5.3. Resultados experimentales

En esta sección se presentan las tablas comparativas, resultado de la experimentación. Siguiendo los criterios mencionados en las secciones anteriores, cada renglón representa un POM, y cada columna representa un AEMO. La comparativa consiste en mostrar la media y desviación estándar de cada 30 ejecuciones por algoritmo y por problema de prueba. En la comparativa se muestra con color oscuro el mejor resultado para dicho POM, en color un poco menos obscuro se muestra el segundo lugar, y los lugares siguientes se muestran sin color.

Tabla 5.5: Hipervolumen obtenido por los algoritmos DMOPSO11, MOEAD, SMPSO, dMOPSO. Se muestra la media y desviación estándar de 30 ejecuciones independientes.

$MOP(objs)$	DMOPSO11 $media_\sigma$	MOEAD $media_\sigma$	SMPSO $media_\sigma$	dMOPSO $media_\sigma$
ZDT1(2)	$0.870641_{0.000053}$	$0.831847_{0.012772}$	$0.871684_{0.000163}$	$0.870305_{0.000112}$
ZDT2(2)	$0.537551_{0.000118}$	$0.494007_{0.014351}$	$0.538528_{0.000088}$	$0.466447_{0.159407}$
ZDT3(2)	$1.325267_{0.000864}$	$1.140106_{0.058229}$	$1.321293_{0.021988}$	$1.322173_{0.000741}$
ZDT4(2)	$0.863952_{0.017279}$	$0.314566_{0.196986}$	$0.871302_{0.000190}$	$0.666190_{0.335396}$
ZDT6(2)	$0.503047_{0.000291}$	$0.503911_{0.002135}$	$0.504427_{0.000063}$	$0.504521_{0.000000}$
DTLZ1(3)	$1.133617_{0.215363}$	$1.177359_{0.171492}$	$1.274530_{0.040679}$	$1.249186_{0.020505}$
DTLZ2(3)	$7.398277_{0.002588}$	$7.391139_{0.002021}$	$7.357531_{0.009388}$	$7.342886_{0.009425}$
DTLZ3(3)	$330.648285_{20.048380}$	$325.356903_{39.676968}$	$339.569641_{10.460958}$	$339.583496_{2.325715}$
DTLZ4(3)	$7.399940_{0.002736}$	$7.380946_{0.033694}$	$7.365068_{0.012247}$	$7.317987_{0.018891}$
DTLZ5(3)	$59.848194_{0.002602}$	$59.551792_{0.028151}$	$59.871578_{0.000882}$	$59.086609_{0.122005}$
DTLZ6(3)	$1318.197754_{0.262310}$	$1315.700073_{0.026053}$	$1318.952881_{0.604887}$	$1312.181274_{0.098589}$
DTLZ7(3)	$1.530219_{0.328318}$	$1.343062_{0.214468}$	$1.759633_{0.022443}$	$1.705316_{0.007326}$
WFG1(3)	$12.250358_{0.286142}$	$14.122287_{0.160757}$	$13.681060_{0.117514}$	$14.104010_{0.192318}$
WFG2(3)	$43.186367_{0.318696}$	$43.923073_{0.661161}$	$41.987057_{0.581168}$	$41.782391_{0.449564}$
WFG3(3)	$28.323357_{0.194113}$	$27.890774_{0.542063}$	$26.853992_{0.495140}$	$27.285551_{0.396714}$
WFG4(3)	$19.241030_{0.171245}$	$17.560511_{0.461082}$	$17.670675_{0.350208}$	$19.045763_{0.271522}$
WFG5(3)	$20.717487_{0.154631}$	$19.981394_{0.158174}$	$17.491659_{0.678050}$	$18.595472_{0.375606}$
WFG6(3)	$20.860415_{0.427278}$	$18.664070_{0.505660}$	$20.082064_{0.602030}$	$18.768929_{0.615580}$
WFG7(3)	$19.668215_{0.219730}$	$18.383413_{0.623228}$	$15.206646_{0.585685}$	$15.329456_{0.360951}$
WFG8(3)	$15.919049_{0.208065}$	$12.716968_{0.609613}$	$11.601747_{0.444535}$	$11.781309_{0.378755}$
WFG9(3)	$20.647726_{0.342840}$	$18.166241_{0.359171}$	$18.928877_{0.708821}$	$18.590422_{0.557479}$

En problemas con dos objetivos podemos ver que DMOPSO11 obtuvo la mejor convergencia en ZDT3 de acuerdo a los indicadores I_{HV} e I_{IGD+} (ver tablas 5.5 y 5.6). Este problema tiene un frente de Pareto discontinuo y convexo. En los problemas restantes del conjunto ZDT, DMOPSO11 se muestra competitivo en cuanto a convergencia. En cuanto a la distribución, el indicador I_S muestra que DMOPSO11 no tiene un buen desempeño comparado contra los otros AEMOs. Sin embargo, en ZDT3 es superior que dMOPSO y MOEA/D, aunque es superado por SMPSO.

En problemas con tres objetivos tenemos más variabilidad, al tener más instancias disponibles. Comenzaremos con la convergencia en los problemas del conjunto DTLZ. En este caso, podemos decir que el desempeño se muestra mejor que otros AEMOs al resolver DTLZ4. Este problema tiene un frente de Pareto cóncavo, separable y multimodal. Continuando con la convergencia, se puede decir que DMOPSO11 es competitivo en DTLZ1-3,DTLZ5 y DTLZ6 siendo en algunos casos el mejor, y en otros el segundo mejor. Sin embargo, en DTLZ7 no logra una buena convergencia en comparación con los otros AEMOs. En cuanto a la distribución se puede ver que DMOPSO11 es competitivo en DTLZ4-7. En cuanto a los problemas del conjunto WFG, podemos ver que a DMOPSO11 le va muy bien con excepción de WFG1 que es un problema en el cual las soluciones tienen una distribución polinomial y algunas regiones planas. En problemas con tres objetivos podemos ver que DMOPSO11 es competitivo en cuanto a convergencia, aunque no logra un buen desempeño en su distribución de soluciones a lo largo del frente de Pareto.

Tabla 5.6: IGD+ obtenido por los algoritmos DMOPSO11, MOEAD, SMPSO, dMOPSO. Se muestra la media y desviación estándar de 30 ejecuciones independientes.

$MOP(objs)$	DMOPSO11 $media_\sigma$	MOEAD $media_\sigma$	SMPSO $media_\sigma$	dMOPSO $media_\sigma$
ZDT1(2)	$0.002420_{0.000014}$	$0.026556_{0.008198}$	$0.002577_{0.000085}$	$0.002560_{0.000054}$
ZDT2(2)	$0.002379_{0.000014}$	$0.022465_{0.007451}$	$0.002302_{0.000046}$	$0.057916_{0.123551}$
ZDT3(2)	$0.002122_{0.000904}$	$0.087787_{0.030168}$	$0.002792_{0.003405}$	$0.003765_{0.000125}$
ZDT4(2)	$0.004344_{0.012396}$	$0.518192_{0.276710}$	$0.002793_{0.000103}$	$0.180436_{0.293198}$
ZDT6(2)	$0.002156_{0.000018}$	$0.002508_{0.001595}$	$0.002091_{0.000045}$	$0.002081_{0.000000}$
DTLZ1(3)	$0.003642_{0.007460}$	$4.759015_{6.692617}$	$0.029986_{0.022419}$	$0.268519_{1.296567}$
DTLZ2(3)	$0.028979_{0.000718}$	$0.028958_{0.000489}$	$0.041686_{0.003794}$	$0.032870_{0.001324}$
DTLZ3(3)	$0.008456_{0.019743}$	$8.213224_{15.182057}$	$0.032052_{0.019029}$	$0.912681_{4.473670}$
DTLZ4(3)	$0.028445_{0.000743}$	$0.033537_{0.010111}$	$0.039525_{0.004317}$	$0.051633_{0.004951}$
DTLZ5(3)	$0.004343_{0.000331}$	$0.003203_{0.000246}$	$0.001872_{0.000107}$	$0.006011_{0.000396}$
DTLZ6(3)	$0.002560_{0.000052}$	$0.002540_{0.000005}$	$0.001595_{0.000352}$	$0.005210_{0.000145}$
DTLZ7(3)	$0.188911_{0.213974}$	$0.225867_{0.190409}$	$0.046593_{0.004230}$	$0.059400_{0.001021}$
WFG1(3)	$1.678243_{0.022170}$	$1.489192_{0.009094}$	$1.520448_{0.006062}$	$1.487877_{0.004866}$
WFG2(3)	$0.067536_{0.103759}$	$0.163185_{0.027729}$	$0.216067_{0.076383}$	$0.258548_{0.034967}$
WFG3(3)	$0.215876_{0.012736}$	$0.237124_{0.033642}$	$0.286472_{0.030900}$	$0.237380_{0.029030}$
WFG4(3)	$0.193736_{0.003971}$	$0.224164_{0.015807}$	$0.234731_{0.045011}$	$0.192853_{0.006990}$
WFG5(3)	$0.149058_{0.004448}$	$0.155916_{0.003801}$	$0.307131_{0.044234}$	$0.219887_{0.016965}$
WFG6(3)	$0.162787_{0.009283}$	$0.202783_{0.012983}$	$0.177802_{0.015696}$	$0.205282_{0.016540}$
WFG7(3)	$0.180566_{0.005772}$	$0.191175_{0.014450}$	$0.330163_{0.020931}$	$0.321797_{0.013809}$
WFG8(3)	$0.291334_{0.005940}$	$0.410519_{0.033855}$	$0.534515_{0.029651}$	$0.521771_{0.026715}$
WFG9(3)	$0.147397_{0.008601}$	$0.212952_{0.010399}$	$0.220502_{0.050084}$	$0.218729_{0.027604}$

Tabla 5.7: Espaciado obtenido por los algoritmos DMOPSO11, MOEAD, SMPSO, dMOPSO. Se muestra la media y desviación estándar de 30 ejecuciones independientes.

$MOP(objs)$	DMOPSO11 $media_\sigma$	MOEAD $media_\sigma$	SMPSO $media_\sigma$	dMOPSO $media_\sigma$
ZDT1(2)	$0.005827_{0.000181}$	$0.010975_{0.000659}$	$0.002511_{0.000186}$	$0.005023_{0.000065}$
ZDT2(2)	$0.004697_{0.000169}$	$0.007768_{0.001349}$	$0.002453_{0.000127}$	$0.003863_{0.001726}$
ZDT3(2)	$0.017002_{0.007336}$	$0.025788_{0.002633}$	$0.005599_{0.003123}$	$0.017550_{0.000221}$
ZDT4(2)	$1.479166_{2.820395}$	$0.103793_{0.270411}$	$0.002775_{0.000201}$	$0.003526_{0.002137}$
ZDT6(2)	$0.003599_{0.000289}$	$0.003355_{0.000404}$	$0.054321_{0.097543}$	$0.003198_{0.000018}$
DTLZ1(3)	$9.858583_{17.583652}$	$0.665691_{0.825877}$	$1.254115_{2.746904}$	$0.046536_{0.122411}$
DTLZ2(3)	$0.080792_{0.059514}$	$0.060999_{0.000966}$	$0.049947_{0.004557}$	$0.038564_{0.002297}$
DTLZ3(3)	$29.691883_{32.615368}$	$0.842444_{1.337411}$	$1.322883_{3.099600}$	$0.136711_{0.365488}$
DTLZ4(3)	$0.059816_{0.033370}$	$0.062295_{0.021831}$	$0.054387_{0.009211}$	$0.112101_{0.059762}$
DTLZ5(3)	$0.008610_{0.000876}$	$0.008781_{0.000182}$	$0.003406_{0.000576}$	$0.195342_{0.008317}$
DTLZ6(3)	$0.008466_{0.000239}$	$0.008678_{0.000055}$	$0.008421_{0.027945}$	$0.174108_{0.008636}$
DTLZ7(3)	$0.078974_{0.029249}$	$0.081097_{0.026975}$	$0.075380_{0.010545}$	$0.156642_{0.002784}$
WFG1(3)	$0.463078_{0.032008}$	$0.162096_{0.027766}$	$0.163912_{0.016085}$	$0.110558_{0.030272}$
WFG2(3)	$0.199231_{0.041542}$	$0.139181_{0.024542}$	$0.199120_{0.026896}$	$0.150021_{0.015228}$
WFG3(3)	$0.130162_{0.006233}$	$0.148098_{0.007977}$	$0.119469_{0.009401}$	$0.176741_{0.006913}$
WFG4(3)	$0.275763_{0.017682}$	$0.301899_{0.010604}$	$0.200565_{0.020043}$	$0.209459_{0.006101}$
WFG5(3)	$0.252125_{0.012697}$	$0.260588_{0.005665}$	$0.174587_{0.014003}$	$0.226986_{0.011530}$
WFG6(3)	$0.285362_{0.013421}$	$0.260973_{0.012721}$	$0.188597_{0.017177}$	$0.214119_{0.008546}$
WFG7(3)	$0.265750_{0.017032}$	$0.285695_{0.008634}$	$0.182475_{0.014732}$	$0.216197_{0.009045}$
WFG8(3)	$0.261761_{0.013097}$	$0.281793_{0.011302}$	$0.181228_{0.014332}$	$0.224374_{0.011319}$
WFG9(3)	$0.255575_{0.011560}$	$0.258363_{0.009666}$	$0.180801_{0.026298}$	$0.236046_{0.016295}$

En la tabla 5.5 se muestra la comparativa en cuanto al estudio de escalabilidad de dos

a 10 objetivos utilizando en el problema DTLZ2. Es notable en la sección de hipervolumen que DMOPSO11 escala muy bien, siendo mejor que los otros AEMOs. Sin embargo, como no logra ser competitivo en cuanto a la distribución de las soluciones, esto afecta al indicador IGD+.

En la figura 5.1 se pueden visualizar los resultados obtenidos por los AEMOs considerados para el experimento. Para cada imagen se seleccionó la mediana de las 30 ejecuciones independientes realizadas con respecto al hipervolumen. Se procedió de esta misma manera para los resultados mostrados en las figuras 5.2, 5.3, 5.4 y 5.5.

Tabla 5.8: Hipervolumen, IGD+ y espaciado obtenido por los algoritmos DMOPSO11, MOEAD, SMPSO, dMOPSO en DTLZ2 de 2 a 10 objetivos. Se muestra la media y desviación estándar de 30 ejecuciones independientes.

$MOP(objs)$	DMOPSO11 $media_\sigma$	MOEAD $media_\sigma$	SMPSO $media_\sigma$	dMOPSO $media_\sigma$
		Hipervolumen		
DTLZ2(2)	$3.207206_{0.000317}$	$3.209677_{0.000065}$	$3.210269_{0.000123}$	$3.207678_{0.000388}$
DTLZ2(3)	$7.398277_{0.002588}$	$7.391139_{0.002021}$	$7.357531_{0.009388}$	$7.342886_{0.009425}$
DTLZ2(4)	$14.971848_{0.016446}$	$14.537100_{0.007139}$	$14.991097_{0.111157}$	$14.754835_{0.035414}$
DTLZ2(5)	$29.970825_{0.131786}$	$28.749273_{0.025601}$	$27.284252_{1.016019}$	$29.449005_{0.083967}$
DTLZ2(6)	$59.633732_{0.395638}$	$56.939182_{0.070610}$	$48.334965_{3.352436}$	$58.467556_{0.221227}$
DTLZ2(7)	$119.333694_{0.506290}$	$113.697678_{0.181045}$	$93.262199_{6.069544}$	$116.991821_{0.314710}$
DTLZ2(8)	$235.250076_{1.605896}$	$223.886414_{0.610021}$	$127.610077_{33.764572}$	$229.696838_{1.619064}$
DTLZ2(9)	$472.886169_{2.892513}$	$448.182098_{0.914957}$	$211.820053_{63.987396}$	$459.666229_{2.449796}$
DTLZ2(10)	$948.933777_{8.445759}$	$895.292419_{1.879484}$	$367.637573_{149.924210}$	$918.953430_{4.741397}$
		IGD+		
DTLZ2(2)	$0.003605_{0.000165}$	$0.002344_{0.000033}$	$0.002103_{0.000065}$	$0.003400_{0.000204}$
DTLZ2(3)	$0.028979_{0.000718}$	$0.028958_{0.000489}$	$0.041686_{0.003794}$	$0.032870_{0.001324}$
DTLZ2(4)	$0.091384_{0.003028}$	$0.132174_{0.001035}$	$0.183256_{0.033408}$	$0.098198_{0.004060}$
DTLZ2(5)	$0.140412_{0.006615}$	$0.186809_{0.001212}$	$0.387164_{0.150090}$	$0.145591_{0.007310}$
DTLZ2(6)	$0.182170_{0.012449}$	$0.219062_{0.002664}$	$0.587608_{0.145105}$	$0.181166_{0.011027}$
DTLZ2(7)	$0.100262_{0.300787}$	$0.100007_{0.300020}$	$0.114177_{0.342851}$	$0.100000_{0.300000}$
DTLZ2(8)	$0.264169_{0.020891}$	$0.261234_{0.005411}$	$0.892756_{0.255528}$	$0.236364_{0.031819}$
DTLZ2(9)	$0.269018_{0.021476}$	$0.268156_{0.003840}$	$1.030179_{0.180688}$	$0.240628_{0.024891}$
DTLZ2(10)	$0.257942_{0.017117}$	$0.274569_{0.003103}$	$0.976567_{0.355803}$	$0.241264_{0.024638}$
		Espaciado		
DTLZ2(2)	$0.007675_{0.000682}$	$0.006593_{0.000137}$	$0.002726_{0.000232}$	$0.006015_{0.000253}$
DTLZ2(3)	$0.080792_{0.059514}$	$0.060999_{0.000966}$	$0.049947_{0.004557}$	$0.038564_{0.002297}$
DTLZ2(4)	$0.125356_{0.062052}$	$0.078821_{0.002159}$	$0.160957_{0.014963}$	$0.055251_{0.003813}$
DTLZ2(5)	$0.118984_{0.045925}$	$0.093963_{0.002671}$	$0.344943_{0.026637}$	$0.073158_{0.005853}$
DTLZ2(6)	$0.133094_{0.034051}$	$0.107912_{0.004168}$	$0.513462_{0.031671}$	$0.095741_{0.006467}$
DTLZ2(7)	$0.121594_{0.020602}$	$0.095373_{0.002467}$	$0.588499_{0.033606}$	$0.094480_{0.005126}$
DTLZ2(8)	$0.206750_{0.050080}$	$0.131358_{0.005526}$	$0.831883_{0.049327}$	$0.136300_{0.008470}$
DTLZ2(9)	$0.188143_{0.033086}$	$0.125401_{0.003894}$	$0.920527_{0.045817}$	$0.133532_{0.010456}$
DTLZ2(10)	$0.190840_{0.027874}$	$0.116785_{0.003440}$	$1.007550_{0.043875}$	$0.128662_{0.005541}$

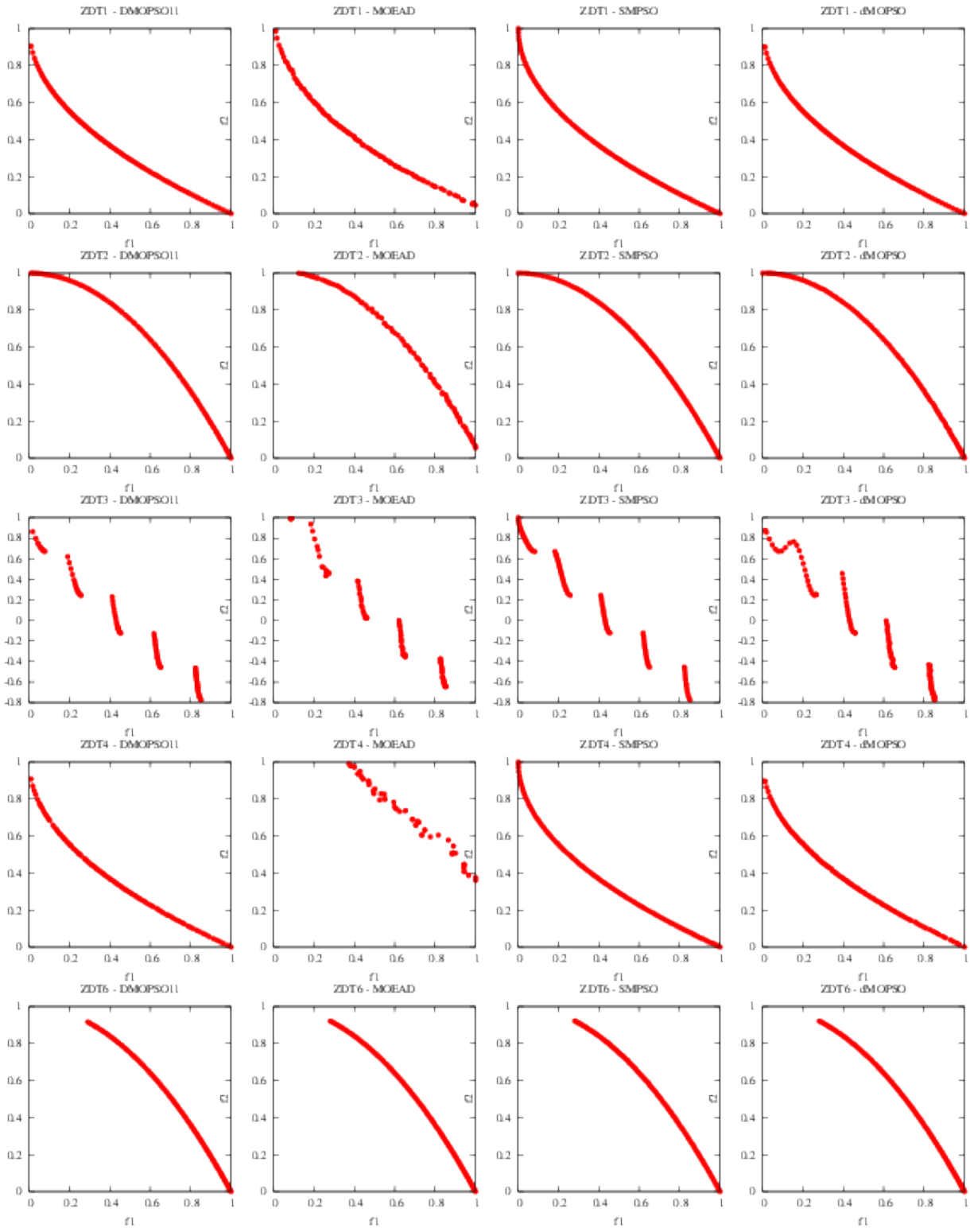

Figura 5.1: Soluciones obtenidas por los algoritmos DMOPSO11, MOEA/D, SMPSO y dMOPSO en los problemas ZDT.

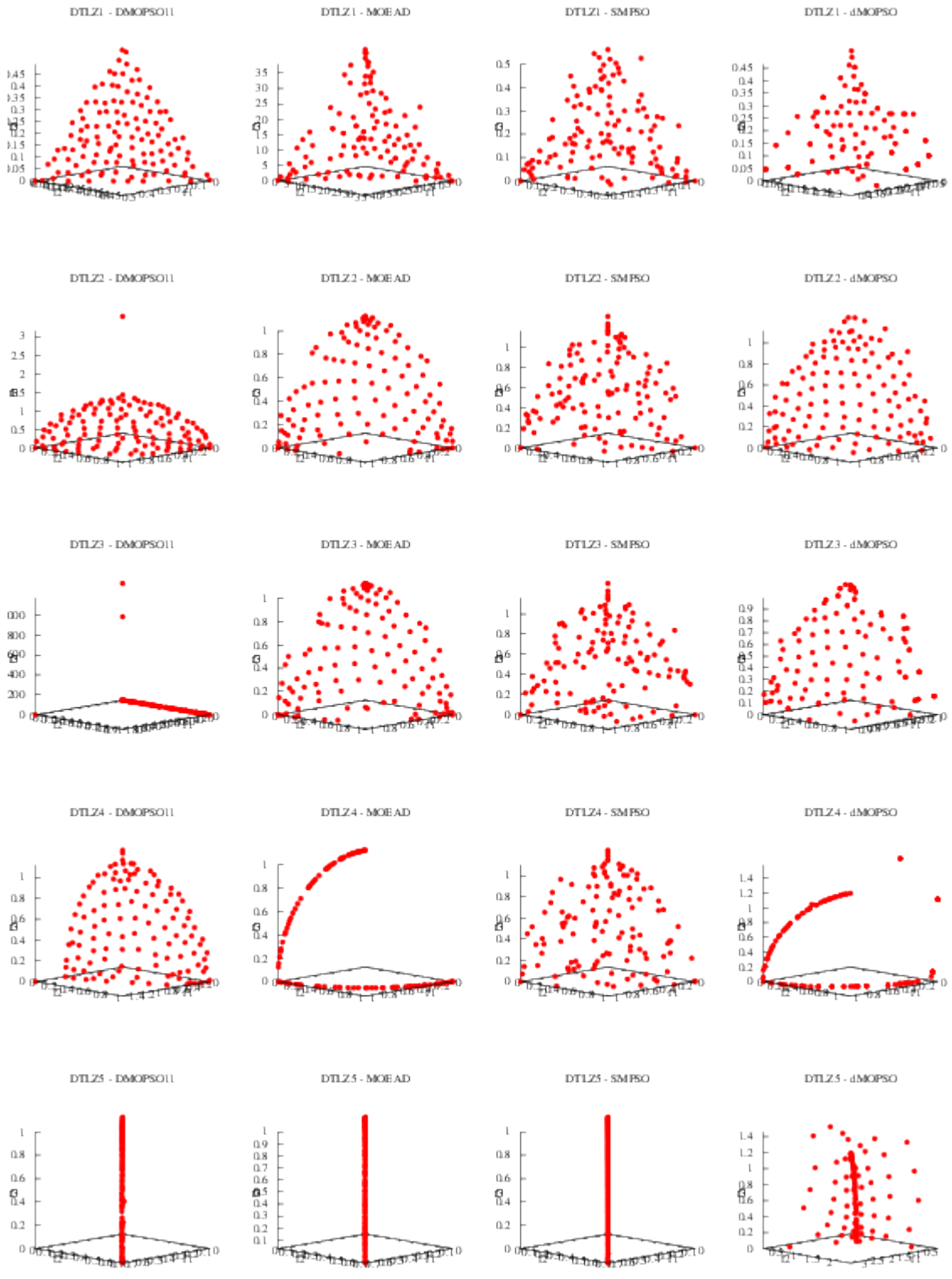

Figura 5.2: Soluciones obtenidas por los algoritmos DMOPSO11, MOEA/D, SMPSO y dMOPSO en DTLZ1, DTLZ2, DTLZ3, DTLZ4 y DTLZ5 con tres objetivos.

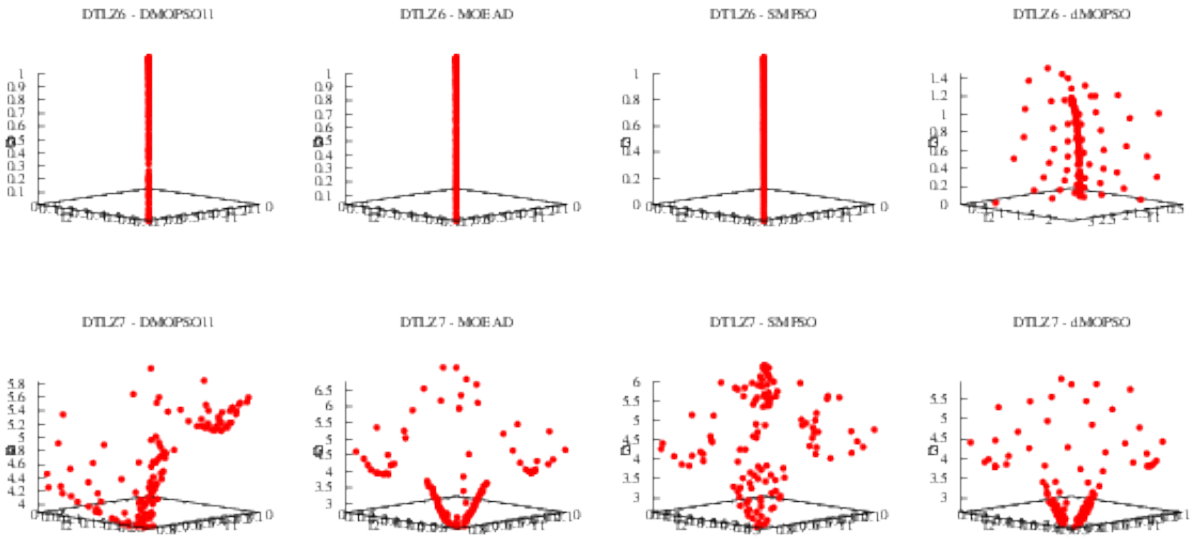

Figura 5.3: Soluciones obtenidas por los algoritmos DMOPSO11, MOEA/D, SMPSO y dMOPSO en DTLZ6 y DTLZ7 con tres objetivos.

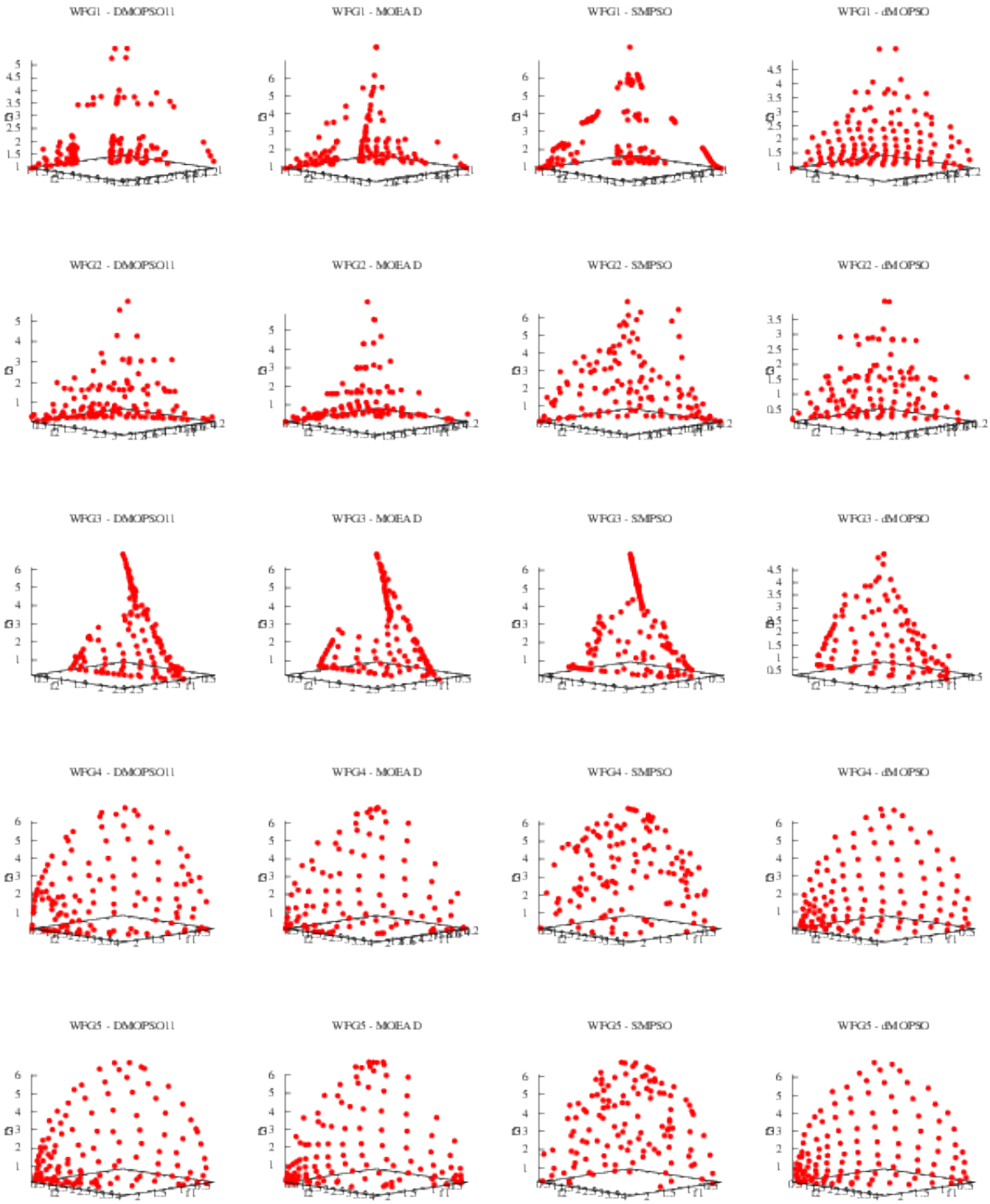

Figura 5.4: Soluciones obtenidas por los algoritmos DMOPSO11, MOEA/D, SMPSO y dMOPSO en WFG1, WFG2, WFG3, WFG4 y WFG5 con tres objetivos.

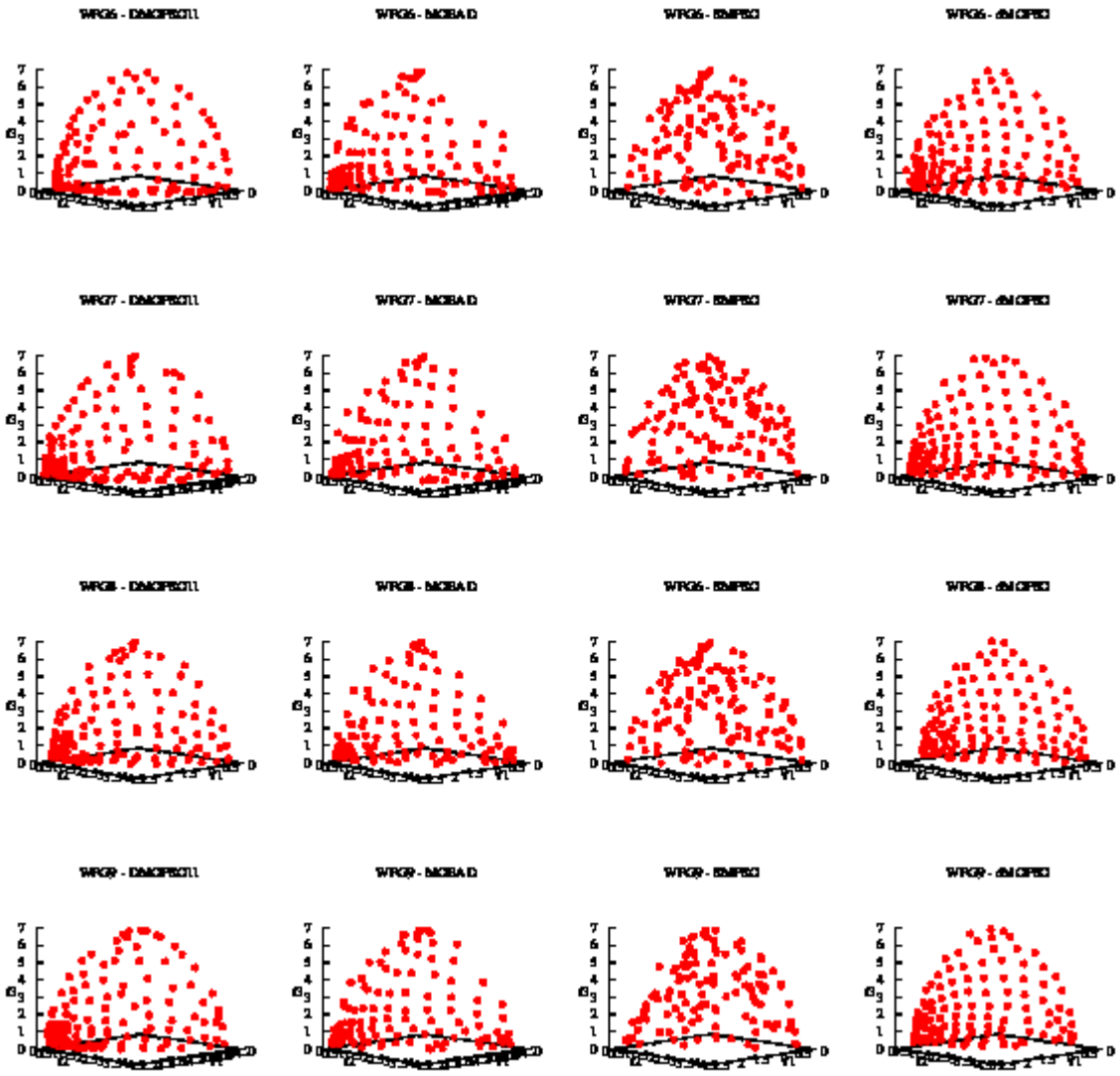

Figura 5.5: Soluciones obtenidas por los algoritmos DMOPSO11, MOEA/D, SMPSO y dMOPSO en WFG6, WFG7, WFG8 y WFG9 con tres objetivos.

Además, se realiza un estudio de significancia estadística para validar los resultados obtenidos, empleando la prueba de la suma de rango de Wilcoxon con un nivel significativo del 5 %. Estos resultados se muestran en las tablas 5.9, 5.12, 5.10, 5.13, 5.11 y 5.14. Es importante mostrar este estudio para saber que tan validos son los resultados de nuestro experimento comparativo, por ejemplo, en la tabla 5.9 que corresponde a los resultados de la tabla 5.5 podemos ver que la columna completa de MOEA/D se muestra H = 1, esto significa que hay significancia estadística en los resultados y podemos confiar en que la comparación de los resultados de MOEA/D y DMOPSO en la tabla 5.5 son confiables.

Tabla 5.9: Análisis estadístico utilizando la suma de rango de Wilcoxon para ZDT, DTLZ y WFG en las evaluaciones de hipervolumen con respecto a los resultados de DMOPSO11. Un valor pequeño de P pone en duda la validez de la hipótesis nula. H = 1 implica que la hipótesis nula puede ser rechazada con un nivel significativo del 5 % y, por lo tanto, hay significancia estadística en los resultados.

MOP(objs)	MOEA/D P(H)	SMPSO P(H)	dMOPSO P(H)
ZDT1(2)	0.000 (1)	0.000 (1)	0.000 (1)
ZDT2(2)	0.000 (1)	0.000 (1)	0.000 (1)
ZDT3(2)	0.000 (1)	0.900 (0)	0.000 (1)
ZDT4(2)	0.000 (1)	0.000 (1)	0.139 (0)
ZDT6(2)	0.000 (1)	0.000 (1)	0.000 (1)
DTLZ1(3)	0.000 (1)	0.000 (1)	0.000 (1)
DTLZ2(3)	0.000 (1)	0.000 (1)	0.000 (1)
DTLZ3(3)	0.000 (1)	0.000 (1)	0.000 (1)
DTLZ4(3)	0.000 (1)	0.000 (1)	0.000 (1)
DTLZ5(3)	0.000 (1)	0.000 (1)	0.000 (1)
DTLZ6(3)	0.000 (1)	0.000 (1)	0.000 (1)
DTLZ7(3)	0.000 (1)	0.002 (1)	0.000 (1)
WFG1(3)	0.000 (1)	0.000 (1)	0.000 (1)
WFG2(3)	0.000 (1)	0.000 (1)	0.002 (1)
WFG3(3)	0.008 (1)	0.000 (1)	0.000 (1)
WFG4(3)	0.000 (1)	0.000 (1)	0.000 (1)
WFG5(3)	0.000 (1)	0.013 (1)	0.000 (1)
WFG6(3)	0.000 (1)	0.000 (1)	0.000 (1)
WFG7(3)	0.000 (1)	0.018 (1)	0.000 (1)
WFG8(3)	0.000 (1)	0.001 (1)	0.000 (1)
WFG9(3)	0.000 (1)	0.001 (1)	0.000 (1)

Tabla 5.10: Análisis estadístico utilizando la suma de rango de Wilcoxon para ZDT, DTLZ y WFG en las evaluaciones de IGD+ con respecto a los resultados de DMOPSO11. Un valor pequeño de P pone en duda la validez de la hipótesis nula. H = 1 implica que la hipótesis nula puede ser rechazada con un nivel significativo del 5 % y, por lo tanto, hay significancia estadística en los resultados.

MOP(objs)	MOEA/D P(H)	SMPSO P(H)	dMOPSO P(H)
ZDT1(2)	0.000 (1)	0.000 (1)	0.000 (1)
ZDT2(2)	0.000 (1)	0.000 (1)	0.000 (1)
ZDT3(2)	0.000 (1)	0.000 (1)	0.000 (1)
ZDT4(2)	0.000 (1)	0.000 (1)	0.701 (0)
ZDT6(2)	0.000 (1)	0.000 (1)	0.000 (1)
DTLZ1(3)	0.000 (1)	0.000 (1)	0.000 (1)
DTLZ2(3)	0.000 (1)	0.000 (1)	0.000 (1)
DTLZ3(3)	0.000 (1)	0.000 (1)	0.000 (1)
DTLZ4(3)	0.000 (1)	0.520 (0)	0.000 (1)
DTLZ5(3)	0.000 (1)	0.000 (1)	0.000 (1)
DTLZ6(3)	0.000 (1)	0.000 (1)	0.000 (1)
DTLZ7(3)	0.000 (1)	0.000 (1)	0.000 (1)
WFG1(3)	0.000 (1)	0.027 (1)	0.000 (1)
WFG2(3)	0.000 (1)	0.000 (1)	0.038 (1)
WFG3(3)	0.000 (1)	0.000 (1)	0.000 (1)
WFG4(3)	0.888 (0)	0.000 (1)	0.000 (1)
WFG5(3)	0.246 (0)	0.009 (1)	0.000 (1)
WFG6(3)	0.000 (1)	0.000 (1)	0.000 (1)
WFG7(3)	0.000 (1)	0.001 (1)	0.000 (1)
WFG8(3)	0.000 (1)	0.026 (1)	0.000 (1)
WFG9(3)	0.000 (1)	0.000 (1)	0.000 (1)

Tabla 5.11: Análisis estadístico utilizando la suma de rango de Wilcoxon para ZDT, DTLZ y WFG en las evaluaciones de Espaciado con respecto a los resultados de DMOPSO11. Un valor pequeño de P pone en duda la validez de la hipótesis nula. H = 1 implica que la hipótesis nula puede ser rechazada con un nivel significativo del 5 % y, por lo tanto, hay significancia estadística en los resultados.

MOP(objs)	MOEA/D P(H)	SMPSO P(H)	dMOPSO P(H)
ZDT1(2)	0.000 (1)	0.000 (1)	0.000 (1)
ZDT2(2)	0.000 (1)	0.000 (1)	0.000 (1)
ZDT3(2)	0.000 (1)	0.000 (1)	0.000 (1)
ZDT4(2)	0.000 (1)	0.000 (1)	0.000 (1)
ZDT6(2)	0.000 (1)	0.186 (0)	0.000 (1)
DTLZ1(3)	0.000 (1)	0.000 (1)	0.000 (1)
DTLZ2(3)	0.000 (1)	0.000 (1)	0.000 (1)
DTLZ3(3)	0.000 (1)	0.001 (1)	0.000 (1)
DTLZ4(3)	0.000 (1)	0.751 (0)	0.000 (1)
DTLZ5(3)	0.000 (1)	0.000 (1)	0.000 (1)
DTLZ6(3)	0.000 (1)	0.000 (1)	0.000 (1)
DTLZ7(3)	0.000 (1)	0.000 (1)	0.000 (1)
WFG1(3)	0.000 (1)	0.000 (1)	0.001 (1)
WFG2(3)	0.000 (1)	0.000 (1)	0.000 (1)
WFG3(3)	0.000 (1)	0.000 (1)	0.000 (1)
WFG4(3)	0.000 (1)	0.000 (1)	0.000 (1)
WFG5(3)	0.000 (1)	0.000 (1)	0.000 (1)
WFG6(3)	0.706 (0)	0.000 (1)	0.000 (1)
WFG7(3)	0.000 (1)	0.000 (1)	0.000 (1)
WFG8(3)	0.000 (1)	0.000 (1)	0.000 (1)
WFG9(3)	0.000 (1)	0.000 (1)	0.000 (1)

Tabla 5.12: Análisis estadístico utilizando la suma de rango de Wilcoxon para DTLZ2 de 2 a 10 objetivos en las evaluaciones de hipervolumen con respecto a los resultados de DMOPSO11. Un valor pequeño de P pone en duda la validez de la hipótesis nula. H = 1 implica que la hipótesis nula puede ser rechazada con un nivel significativo del 5 % y, por lo tanto, hay significancia estadística en los resultados.

MOP(objs)	MOEA/D P(H)	SMPSO P(H)	dMOPSO P(H)
DTLZ2(2)	0.000 (1)	0.000 (1)	0.000 (1)
DTLZ2(3)	0.000 (1)	0.000 (1)	0.000 (1)
DTLZ2(4)	0.000 (1)	0.176 (0)	0.000 (1)
DTLZ2(5)	0.000 (1)	0.000 (1)	0.000 (1)
DTLZ2(6)	0.000 (1)	0.000 (1)	0.000 (1)
DTLZ2(7)	0.000 (1)	0.000 (1)	0.000 (1)
DTLZ2(8)	0.000 (1)	0.000 (1)	0.000 (1)
DTLZ2(9)	0.000 (1)	0.000 (1)	0.000 (1)
DTLZ2(10)	0.000 (1)	0.000 (1)	0.000 (1)

Tabla 5.13: Análisis estadístico utilizando la suma de rango de Wilcoxon para DTLZ2 de 2 a 10 objetivos en las evaluaciones de IGD+ con respecto a los resultados de DMOPSO11. Un valor pequeño de P pone en duda la validez de la hipótesis nula. H = 1 implica que la hipótesis nula puede ser rechazada con un nivel significativo del 5 % y, por lo tanto, hay significancia estadística en los resultados.

MOP(objs)	MOEA/D P(H)	SMPSO P(H)	dMOPSO P(H)
DTLZ2(2)	0.000 (1)	0.000 (1)	0.000 (1)
DTLZ2(3)	0.516 (0)	0.000 (1)	0.000 (1)
DTLZ2(4)	0.000 (1)	0.000 (1)	0.000 (1)
DTLZ2(5)	0.000 (1)	0.000 (1)	0.005 (1)
DTLZ2(6)	0.000 (1)	0.000 (1)	0.610 (0)
DTLZ2(7)	0.000 (1)	0.000 (1)	0.000 (1)
DTLZ2(8)	0.252 (0)	0.000 (1)	0.000 (1)
DTLZ2(9)	0.784 (0)	0.000 (1)	0.000 (1)
DTLZ2(10)	0.000 (1)	0.000 (1)	0.000 (1)

Tabla 5.14: Análisis estadístico utilizando la suma de rango de Wilcoxon para DTLZ2 de 2 a 10 objetivos en las evaluaciones de espaciado con respecto a los resultados de DMOPSO11. Un valor pequeño de P pone en duda la validez de la hipótesis nula. H = 1 implica que la hipótesis nula puede ser rechazada con un nivel significativo del 5 % y, por lo tanto, hay significancia estadística en los resultados.

MOP(objs)	MOEA/D P(H)	SMPSO P(H)	dMOPSO P(H)
DTLZ2(2)	0.000 (1)	0.000 (1)	0.000 (1)
DTLZ2(3)	0.040 (1)	0.068 (0)	0.000 (1)
DTLZ2(4)	0.379 (0)	0.050 (0)	0.000 (1)
DTLZ2(5)	0.918 (0)	0.000 (1)	0.000 (1)
DTLZ2(6)	0.000 (1)	0.000 (1)	0.000 (1)
DTLZ2(7)	0.000 (1)	0.000 (1)	0.000 (1)
DTLZ2(8)	0.000 (1)	0.000 (1)	0.000 (1)
DTLZ2(9)	0.000 (1)	0.000 (1)	0.000 (1)
DTLZ2(10)	0.000 (1)	0.000 (1)	0.000 (1)

Capítulo 6

Conclusiones y trabajo futuro

Proponer un MOPSO basado en el SPSO2011 resultó ser una tarea complicada, ya que montar el SPSO2011 en el esquema de un MOPSO general no produce un algoritmo superior a otro que adopte un estándar anterior, como el SPSO2007. Por lo tanto, a la hora de diseñar el algoritmo se tiene que tener idea de la capacidad y de las limitantes del motor de búsqueda del SPSO2011.

La propuesta presentada en esta tesis, es el producto de pruebas con diferentes enfoques para llevar a cabo un buen desempeño al utilizar el SPSO 2011 en problemas multi-objetivo. En cuanto a la experimentación realizada, fue necesario utilizar el coeficiente de constricción para ajustar la velocidad y evitar que se tuviera un sesgo hacia los límites de las variables. De lo contrario, el desempeño del algoritmo propuesto se degrada considerablemente. Los resultados obtenidos por nuestra propuesta en problemas con 2 y 3 objetivos, muestran que es competitiva con respecto a MOEA/D, dMOPSO, y SMPSO. Los resultados obtenidos en problemas con 4 a 10 objetivos muestran que, en cuanto a convergencia, DMOPSO11 se desempeña de mejor manera que MOEA/D, dMOPSO, y SMPSO. Sin embargo, en la mayoría de los casos, el algoritmo propuesto tuvo un mal desempeño en lo referente a la distribución de sus soluciones, lo cual indica que se requiere de un mejor estimador de densidad.

En cuanto al trabajo futuro, existe todavía la posibilidad de diseñar un esquema simple e intuitivo que permita aprovechar el SPSO2011 de mejor manera. Para lograrlo, hay que extender este trabajo realizando las siguientes actividades:

- Tomar una mayor cantidad de MOPSOs del estado del arte y cambiarles el motor de búsqueda por el del SPSO2011, para ver sus efectos tanto negativos como positivos. Esto permitirá identificar los factores que hacen que el SPSO2011 tenga un mejor desempeño en algunos problemas, y uno más malo en otros.
- Intentar regular el radio de la hiper-esfera, o implementar alguna técnica que permita ajustar una distribución regulable dentro de la hiper-esfera. Es importante

mencionar esto ya que se descubrió que este factor es muy importante a la hora de atacar un problema en particular.

- Investigar el uso de otros estimadores de densidad que resulten adecuados para el algoritmo propuesto, de tal forma que se mejore la distribución de sus soluciones.

Apéndice A

Problemas de prueba

En este apéndice se describen los tres conjuntos de prueba que suelen usarse más frecuentemente en optimización evolutiva multi-objetivo y que se adoptaron en esta tesis. Estos problemas de prueba examinan la capacidad y desempeño de los optimizadores evolutivos multi-objetivo, ya que proporcionan distintas fuentes de dificultad para alcanzar el verdadero frente de Pareto. De tal forma, estos problemas permiten evaluar la capacidad de un algoritmo evolutivo multi-objetivo para lidiar con distintas características que un problema multi-objetivo puede presentar (p.ej., multi-frontalidad, discontinuidades, concavidades, etc.).

A.1. Conjunto de problemas *Zitzler-Deb-Thiele* (ZDT)

El diseño de estos problemas pone énfasis en algunas de las características que causan dificultades para un algoritmo evolutivo multi-objetivo. Estos problemas tienen una misma estructura y consisten de tres funciones, F, g y h, las cuales se describen a continuación [101]:

$$\text{Minimizar } F = (f_1(x_1), f_2(x_2))$$
$$\text{Sujeto a } f_2(x) = g(x_2, ..., x_m) \cdot h(f_1(x_1), g(x_2, ..., x_m))$$
$$\text{donde } x = (x_1, ..., x_m)$$

La función f_1 depende solamente de la primera variable de decisión, g es una función de las $m-1$ variables restantes, y los parámetros de h son los valores de las funciones f_1 y g.

- La función de prueba ZDT1 tiene un frente de Pareto convexo:

$$f_1(x) = x_1$$

$$f_2(x, g(x)) = g(x) \cdot (1 - \sqrt{\frac{f_1}{g(x)}})$$

$$g(x) = 1 + \frac{9}{n-1} \cdot \sum_{i=2}^{n} x_i$$

Donde $n = 30$ y $x_i \in [0, 1]$ con $i = 1, ..., n$. El frente de Pareto verdadero se forma con $g(x) = 1$.

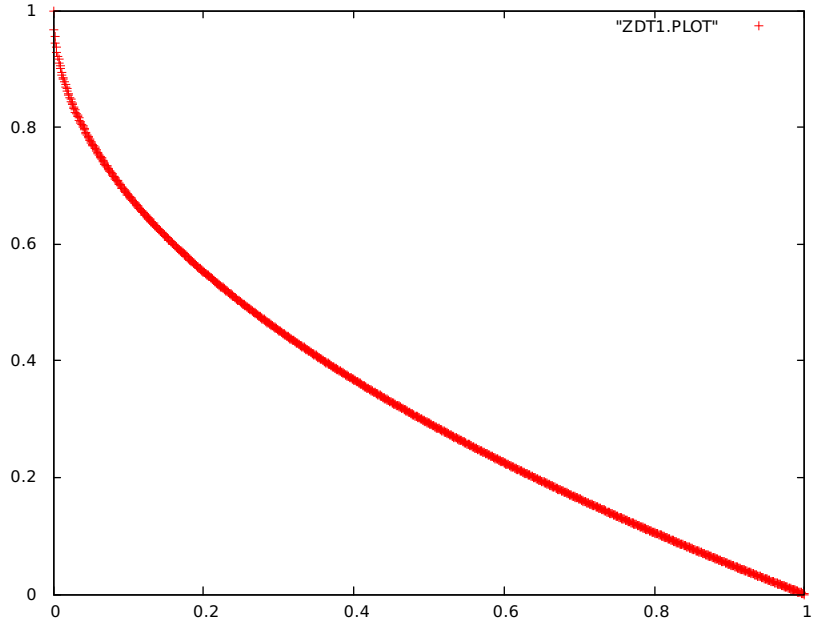

Figura A.1: Frente de Pareto verdadero de ZDT1

- La función de prueba ZDT2 tiene un frente de Pareto cóncavo:

$$f_1(x) = x_1$$

$$f_2(x, g(x)) = g(x) \cdot (1 - (\frac{f_1}{g(x)})^2)$$

$$g(x) = 1 + \frac{9}{n-1} \cdot \sum_{i=2}^{n} x_i$$

Donde $n = 30$ y $x_i \in [0, 1]$ con $i = 1, ..., n$. El frente de Pareto verdadero se forma con $g(x) = 1$.

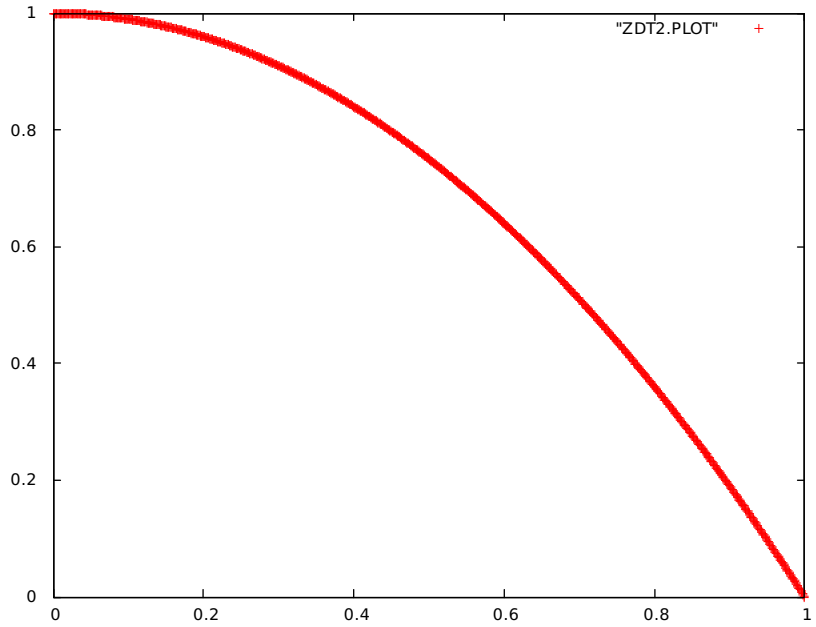

Figura A.2: Frente de Pareto verdadero de ZDT2

- La función de prueba ZDT3 tiene un frente de Pareto cóncavo:

$$f_1(x) = x_1$$

$$f_2(x, g(x)) = g(x) \cdot \left(1 - \sqrt{\frac{f_1(x)}{g(x)}} - \frac{f_1(x)}{g(x)} sen(10 \cdot \pi \cdot f_1(x))\right)$$

$$g(x) = 1 + \frac{9}{n-1} \cdot \sum_{i=2}^{n} x_i$$

Donde $n = 30$ y $x_i \in [0, 1]$ con $i = 1, ..., n$. El frente de Pareto verdadero se forma con $g(x) = 1$. La función seno en h produce una discontinuidad en el frente de Pareto. Sin embargo, no hay discontinuidad en el espacio de las variables de decisión

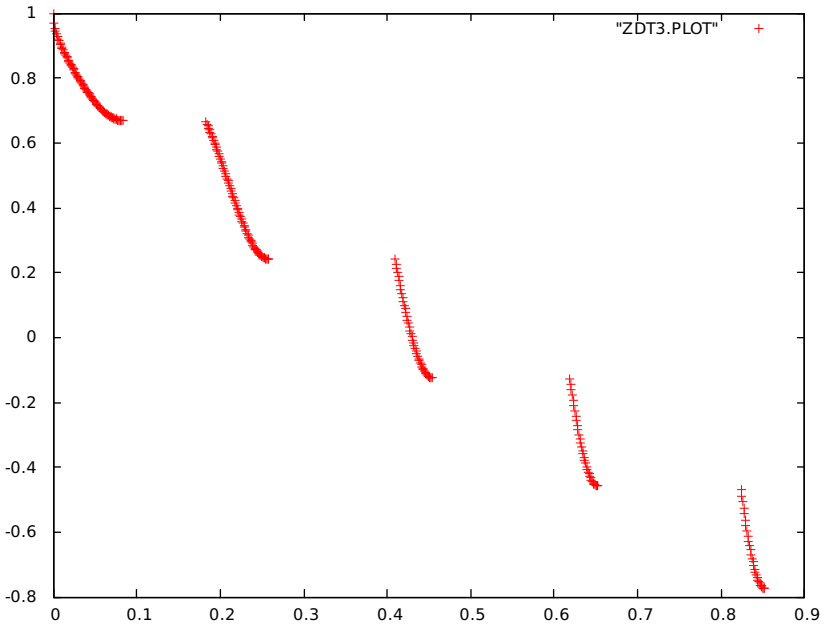

Figura A.3: Frente de Pareto verdadero de ZDT3

- La función de prueba ZDT4 tiene 21^9 frentes locales, lo que pone a prueba la habilidad de un algoritmo para lidiar con problemas multifrontales:

$$f_1(x) = x_1$$

$$f_2(x, g(x)) = g(x) \cdot \left(1 - \sqrt{\frac{f_1(x)}{g(x)}}\right)$$

$$g(x) = 1 + 10 \cdot (n-1) + \sum_{i=2}^{n} (x_i^2 - 10 \cdot cos(4 \cdot \pi \cdot x_i))$$

Donde $n = 10$ y $x_1 \in [0, 1]$ y $x_i \in [-5, 5]$ con $i = 2, ..., n$. El frente de Pareto verdadero se forma con $g(x) = 1$.

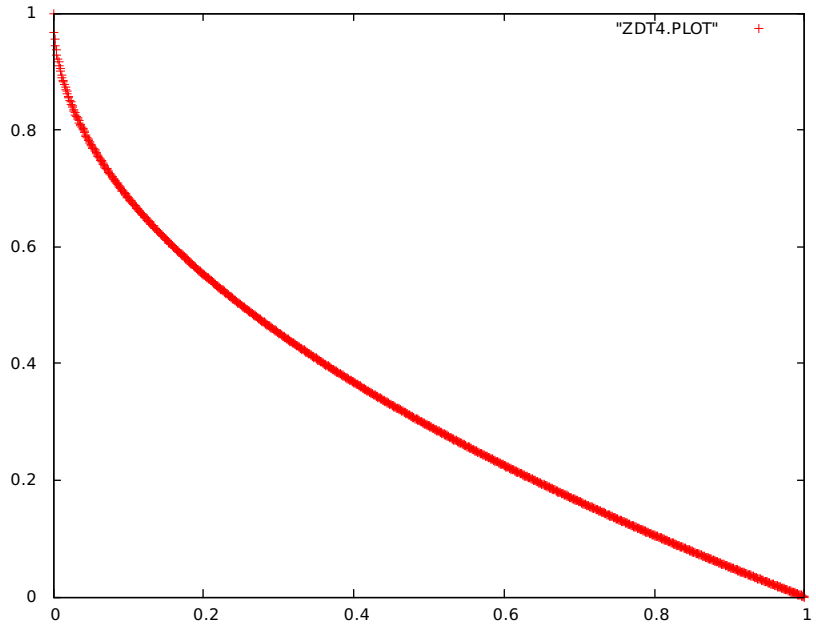

Figura A.4: Frente de Pareto verdadero de ZDT4

- La función de prueba ZDT6 tiene un frente de Pareto cóncavo:

$$f_1(x) = 1 - e^{(-4 \cdot x_1)} \cdot sen^6(6 \cdot \pi \cdot x_1)$$

$$f_2(x, g(x)) = g(x) \cdot \left(1 - \left(\frac{f_1}{g(x)}\right)^2\right)$$

$$g(x) = 1 + 9 \cdot \left[\frac{\sum_{i=2}^{n}}{9}\right]^{0.25}$$

Donde $n = 10$, $x_1 \in [0, 1]$ y $x_i \in [-5, 5]$ con $i = 2, ..., n$. El frente de Pareto verdadero se forma con $g(x) = 1$. Este problema tiene una baja densidad en las soluciones cerca del frente de Pareto verdadero y una alta densidad lejos del mismo.

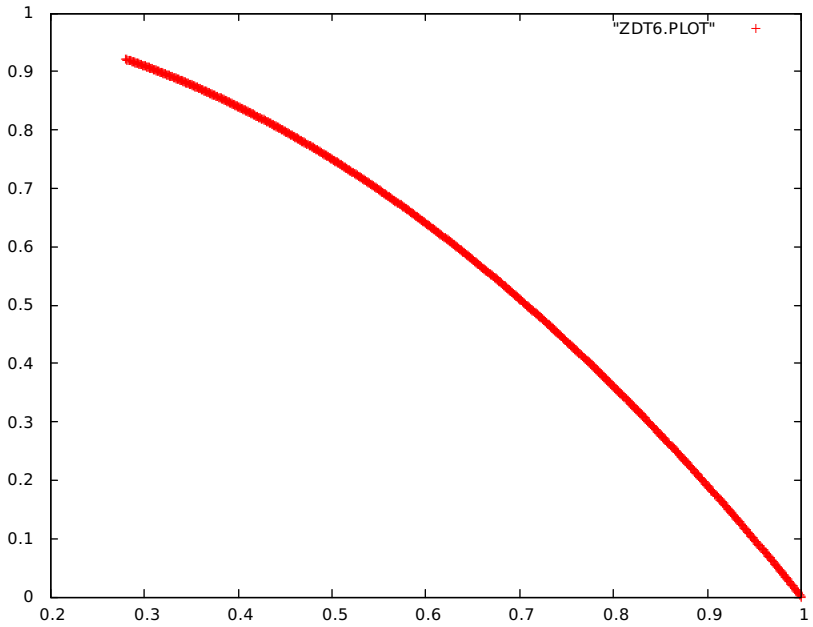

Figura A.5: Frente de Pareto verdadero de ZDT6

A.2. Conjunto de problemas *Deb-Thiele-Laumanns-Zitzler* (DTLZ)

El conjunto de prueba *Deb-Thiele-Laumanns-Zitzler* (DTLZ) incluye nueve problemas para medir el desempeño de los optimizadores evolutivos multi-objetivo. Este conjunto de problemas es escalable a cualquier número de variables de decisión y a cualquier número de objetivos. La mayoría de estos problemas son separables, incluyendo frentes de Pareto multimodales. A continuación, se presentan 7 problemas sin restricciones del conjunto de problemas DTLZ, donde la cantidad total de variables está definida por n, el número de objetivos por m y la cantidad de variables para la función g está definida por k. La función g define la dificultad del problema [102].

- La función de prueba DTLZ1 tiene un frente de Pareto lineal, separable y multimodal:

$$f_1(x) = \tfrac{1}{2} \cdot x_1 \cdot x_2 \cdot \ldots \cdot x_{M-1} \cdot (1 + g(x))$$

$$f_2(x) = \tfrac{1}{2} \cdot x_1 \cdot x_2 \cdot \ldots \cdot (1 - x_{M-1}) \cdot (1 + g(x))$$

$$\vdots$$

$$f_M(x) = \tfrac{1}{2} \cdot (1 - x_1) \cdot (1 + g(x))$$

$$g(x) = 100 \cdot [k + \sum_{i=M}^{n}(x_i - 0.5)^2 - cos(20 \cdot \pi \cdot (x_i - 0.5))]$$

Donde $n = M + k - 1$ (se sugiere una $k = 5$) y $x_i \in [0, 1]$ con $i = 1, ..., n$.

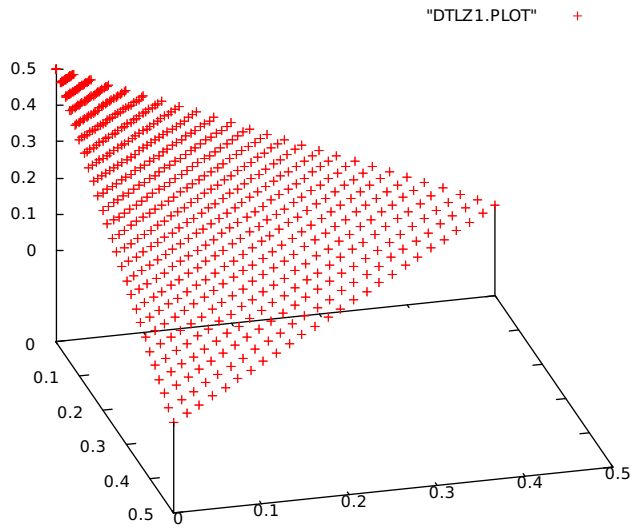

Figura A.6: Frente de Pareto verdadero de DTLZ1 en tres dimensiones

- La función de prueba DTLZ2 tiene un frente de Pareto cóncavo:

$$f_1(x) = cos(x_1\frac{\pi}{2}) \cdot cos(x_2\frac{\pi}{2}) \cdot ... \cdot cos(x_{M-1}\frac{\pi}{2}) \cdot (1 + g(x))$$

$$f_2(x) = cos(x_1\frac{\pi}{2}) \cdot cos(x_2\frac{\pi}{2}) \cdot ... \cdot sen(x_{M-1}\frac{\pi}{2}) \cdot (1 + g(x))$$

$$f_3(x) = cos(x_1\frac{\pi}{2}) \cdot cos(x_2\frac{\pi}{2}) \cdot ... \cdot sen(x_{M-2}\frac{\pi}{2}) \cdot (1 + g(x))$$

$$\vdots$$

$$f_{M-1}(x) = cos(x_1\frac{\pi}{2}) \cdot sen(x_2\frac{\pi}{2})(1 + g(x))$$

$$f_M(x) = sen(x_1\frac{\pi}{2})(1 + g(x))$$

$$g(x) = \sum_i = (x_i - 0.5)^2$$

Donde $n = M + k - 1$ (se sugiere una $k = 10$) y $x_i \in [0, 1]$ con $i = 1, ..., n$. Este problema se puede utilizar para investigar la escalabilidad.

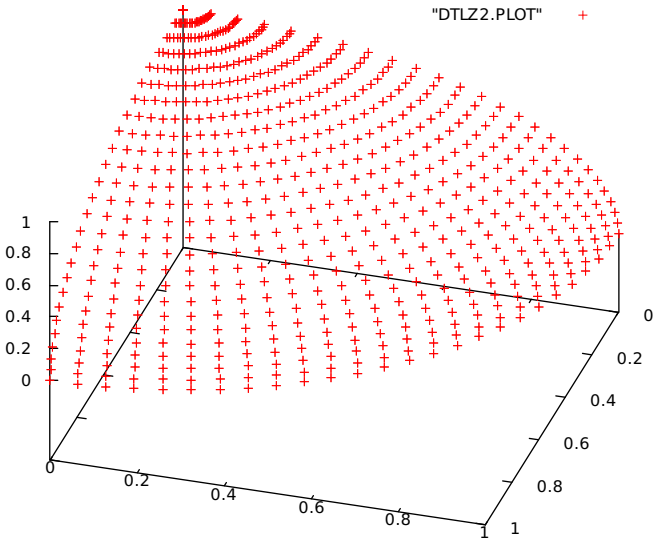

Figura A.7: Frente de Pareto verdadero de DTLZ2 en tres dimensiones

- La función de prueba DTLZ3 tiene un frente de Pareto cóncavo y multimodal:

$$f_1(x) = cos(x_1\frac{\pi}{2}) \cdot cos(x_2\frac{\pi}{2}) \cdot ... \cdot cos(x_{M-1}\frac{\pi}{2}) \cdot (1 + g(x))$$

$$f_2(x) = cos(x_1\frac{\pi}{2}) \cdot cos(x_2\frac{\pi}{2}) \cdot ... \cdot sen(x_{M-1}\frac{\pi}{2}) \cdot (1 + g(x))$$

$$f_3(x) = cos(x_1\frac{\pi}{2}) \cdot cos(x_2\frac{\pi}{2}) \cdot ... \cdot sen(x_{M-2}\frac{\pi}{2}) \cdot (1 + g(x))$$

$$\vdots$$

$$f_{M-1}(x) = cos(x_1\frac{\pi}{2}) \cdot sen(x_2\frac{\pi}{2})(1 + g(x))$$

$$f_M(x) = sen(x_1\frac{\pi}{2})(1 + g(x))$$

$$g(x) = 100 \cdot [k + \textstyle\sum_{i=M}^{n}(x_i - 0.5)^2 - cos(20 \cdot \pi \cdot (x_i - 0.5))]$$

Donde $n = M + k - 1$ (se sugiere una $k = 10$) y $x_i \in [0, 1]$ con $i = 1, ..., n$. La forma del frente de Pareto de este problema es similar a la del problema DTLZ2.

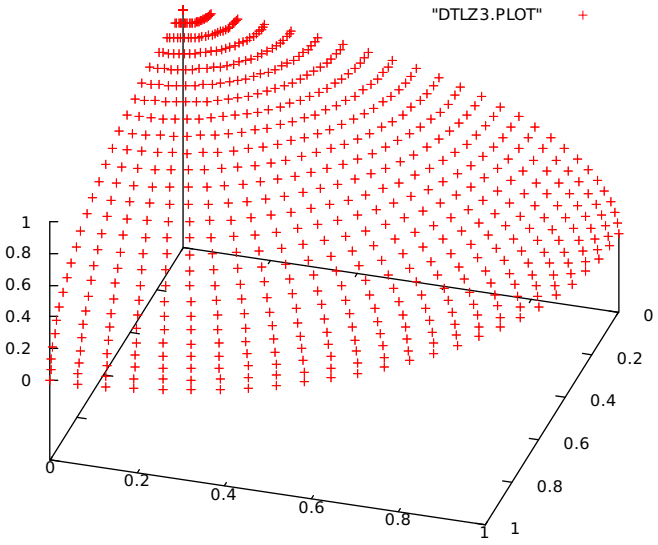

Figura A.8: Frente de Pareto verdadero de DTLZ3 en tres dimensiones

- La función de prueba DTLZ4 tiene un frente de Pareto cóncavo, separable y multimodal:

$$f_1(x) = cos(x_1^\alpha \frac{\pi}{2}) \cdot cos(x_2^\alpha \frac{\pi}{2}) \cdot ... \cdot cos(x_{M-1}^\alpha \frac{\pi}{2}) \cdot (1 + g(x))$$

$$f_2(x) = cos(x_1^\alpha \frac{\pi}{2}) \cdot cos(x_2^\alpha \frac{\pi}{2}) \cdot ... \cdot sen(x_{M-1}^\alpha \frac{\pi}{2}) \cdot (1 + g(x))$$

$$f_3(x) = cos(x_1^\alpha \frac{\pi}{2}) \cdot cos(x_2^\alpha \frac{\pi}{2}) \cdot ... \cdot sen(x_{M-2}^\alpha \frac{\pi}{2}) \cdot (1 + g(x))$$

$$\vdots$$

$$f_{M-1}(x) = cos(x_1^\alpha \frac{\pi}{2}) \cdot sen(x_2^\alpha \frac{\pi}{2})(1 + g(x))$$

$$f_M(x) = sen(x_1^\alpha \frac{\pi}{2})(1 + g(x))$$

$$g(x) = \sum i(x_i - 0.5)^2$$

Donde $n = M + k - 1$ (se sugiere una $k = 10$ y $\alpha = 100$) y $x_i \in [0, 1]$ con $i = 1, ..., n$. Este problema prueba la habilidad de mantener una buena distribución de las soluciones. La forma del frente de Pareto de este problema es similar a la del problema DTLZ2.

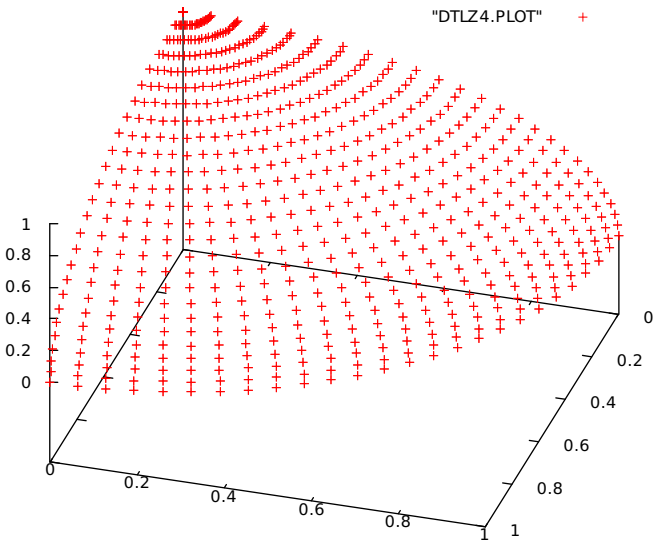

Figura A.9: Frente de Pareto verdadero de DTLZ4 en tres dimensiones

- La función de prueba DTLZ5 tiene un frente de Pareto curvo:

$$f_1(x) = cos(\theta_1 \frac{\pi}{2}) \cdot cos(\theta_2 \frac{\pi}{2}) \cdot ... \cdot cos(\theta_{M-1} \frac{\pi}{2}) \cdot (1 + g(x))$$

$$f_2(x) = cos(\theta_1 \frac{\pi}{2}) \cdot cos(\theta_2 \frac{\pi}{2}) \cdot ... \cdot sen(\theta_{M-1} \frac{\pi}{2}) \cdot (1 + g(x))$$

$$f_3(x) = cos(\theta_1 \frac{\pi}{2}) \cdot cos(\theta_2 \frac{\pi}{2}) \cdot ... \cdot sen(\theta_{M-2} \frac{\pi}{2}) \cdot (1 + g(x))$$

$$\vdots$$

$$f_{M-1}(x) = cos(\theta_1 \frac{\pi}{2}) \cdot sen(\theta_2 \frac{\pi}{2})(1 + g(x))$$

$$f_M(x) = sen(\theta_1 \frac{\pi}{2})(1 + g(x))$$

$$\theta_1 = \frac{\pi}{2} x_1$$

$$\theta_i = \frac{\pi}{4(1+g(x))} \cdot (1 + 2 \cdot g(x) \cdot x_i), \text{ para } i = 2, 3, ..., (M-1)$$

$$g(x) = \sum_i^n (x_i - 0.5)^2$$

Donde $n = M + k - 1$ (se sugiere una $k = 10$) y $x_i \in [0, 1]$ con $i = 1, ..., n$.

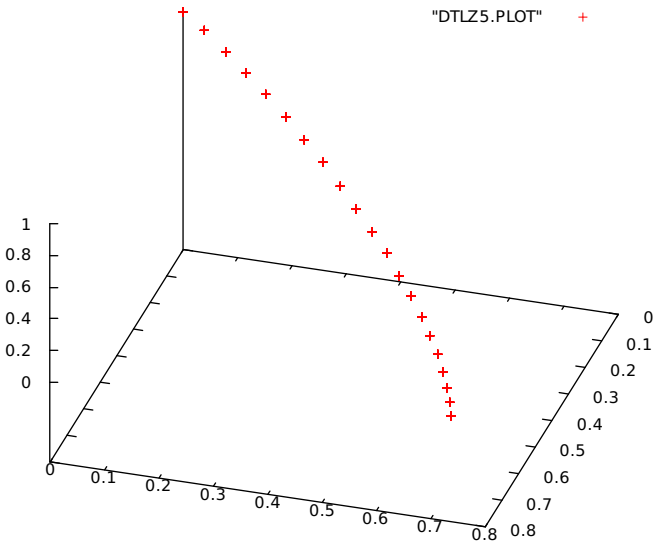

Figura A.10: Frente de Pareto verdadero de DTLZ5 en tres dimensiones

- La función de prueba DTLZ6 tiene un frente de Pareto curvo:

$$f_1(x) = cos(\theta_1 \frac{\pi}{2}) \cdot cos(\theta_2 \frac{\pi}{2}) \cdot ... \cdot cos(\theta_{M-1} \frac{\pi}{2}) \cdot (1 + g(x))$$

$$f_2(x) = cos(\theta_1 \frac{\pi}{2}) \cdot cos(\theta_2 \frac{\pi}{2}) \cdot ... \cdot sen(\theta_{M-1} \frac{\pi}{2}) \cdot (1 + g(x))$$

$$f_3(x) = cos(\theta_1 \frac{\pi}{2}) \cdot cos(\theta_2 \frac{\pi}{2}) \cdot ... \cdot sen(\theta_{M-2} \frac{\pi}{2}) \cdot (1 + g(x))$$

$$\vdots$$

$$f_{M-1}(x) = cos(\theta_1 \frac{\pi}{2}) \cdot sen(\theta_2 \frac{\pi}{2})(1 + g(x))$$

$$f_M(x) = sen(\theta_1 \frac{\pi}{2})(1 + g(x))$$

$$\theta_1 = \frac{\pi}{2} x_1$$

$$\theta_i = \frac{\pi}{4(1+g(x))} \cdot (1 + 2 \cdot g(x) \cdot x_i), \text{ para } i = 2, 3, ..., (M-1)$$

$$g(x) = \sum_i^n (x_i - 0.5)^{0.1}$$

Donde $n = M + k - 1$ (se sugiere una $k = 10$) y $x_i \in [0, 1]$ con $i = 1, ..., n$.

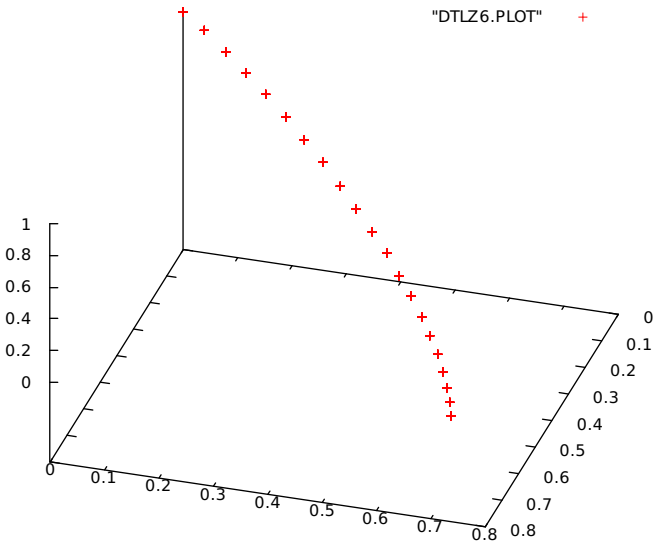

Figura A.11: Frente de Pareto verdadero de DTLZ6 en tres dimensiones

- La función de prueba DTLZ7 tiene un frente de Pareto discontinuo:

$$f_1(x) = x_1$$

$$f_2(x) = x_2$$

$$\vdots$$

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

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

$$g(x) = 1 + \frac{9}{k} \cdot \sum_{i=2}^{n} x_i$$

$$h(f_1, f_2, ..., f_{M-1}g(x)) = M - \sum_{i=1}^{M-1}(\frac{f_i}{1+g(x)}(1 + sen(3 \cdot \pi \cdot f_i)))$$

Donde $n = M + k - 1$ (se sugiere una $k = 20$) y $x_i \in [0, 1]$ con $i = 1, ..., n$. Este problema prueba la habilidad de mantener individuos en diferentes regiones del frente de Pareto.

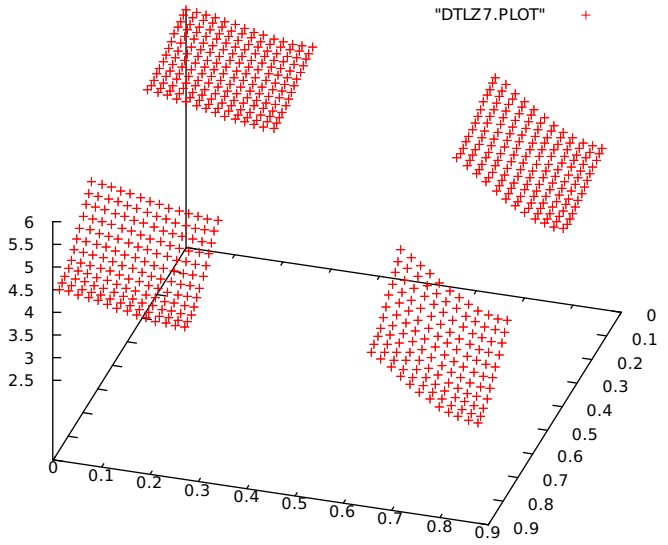

Figura A.12: Frente de Pareto verdadero de DTLZ7 en tres dimensiones

A.3. Conjunto de problemas *Walking-Fish-Group* (WFG)

El conjunto de problemas *Walking-Fish-Group* (WFG) fue propuesto por Huband y Hingston [103]. En su estudio Huband y Hingston propusieron nueve problemas de prueba (WFG1-WFG9), los cuales son escalables con respecto al número de variables y al número de objetivos. Estos problemas tiene una amplia variedad de formas geométricas para los frentes de Pareto. Sin embargo, las características tales como el sesgo, la multi-modalidad y la no separabilidad están definidas por un conjunto de transformaciones. A continuación se muestran las definiciones de nueve problemas del conjunto WFG, donde m representa el número de objetivos, y cada problema es definido en términos de un vector de parámetros.

- El problema de prueba WFG1 es separable y unimodal, pero tiene una región plana y polinomial (ver figura A.13), y se define de la siguiente forma:

  Dado:

  $$\vec{z} = \{z_1, ..., z_k, z_{k+1}, ..., z_n\}$$

  Minimizar:

  $$f_1(\vec{x}) = x_m + 2\prod_{i=1}^{m-1}(1 - cos(x_i\pi/2))$$
  $$f_{j=2:m-1}(\vec{x}) = x_m + 2j\left(\prod_{i=1}^{m-j}(1 - \cos(x_i\pi/2))\right)(1 - \text{sen}(x_{m-j+1}\pi/2))$$
  $$f_m(\vec{x}) = x_m + 2m\left(1 - x_1 - \frac{\cos(10\pi x_1 + \pi/2)}{10\pi}\right)$$

Donde:

$$x_{i=1:m-1} = r\_sum(\{y_{(i-1)k/(m-1)+1}, ..., y_{ik/(m-1)}\}, \{2(i-1)k/(m-1)+1, ..., 2ik/(m-1)\})$$
$$x_m = r\_sum(\{y_{k+1}, ..., y_n\}, \{2(k+1), ..., 2n\})$$
$$y_{i=1:n} = b\_poly(y'_i, 0.02)$$
$$y'_{i=1:k} = y''_i$$
$$y'_{i=k+1:n} = b\_flat(y''_i, 0.8, 0.75, 0.85)$$
$$y''_{i=1:k} = z_i/(2i)$$
$$y''_{i=k+1:n} = s\_linear(z_i/(2i), 0.35)$$

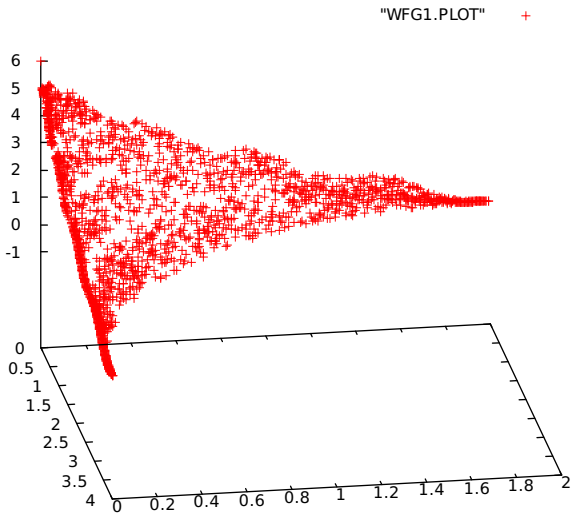

Figura A.13: Frente de Pareto verdadero de WFG1 en tres dimensiones

El problema de prueba WFG2 es no separable y multimodal. Su frente de Pareto verdadero es desconectado (ver figura A.14), y se define de la siguiente forma:

Dado:

$$\vec{z} = \{z_1, ..., z_k, z_{k+1}, ..., z_n\}$$

Minimizar:

$$f_1(\vec{x}) = x_m + 2\prod_{i=1}^{m-1}(1 - cos(x_i\pi/2))$$
$$f_{j=2:m-1}(\vec{x}) = x_m + 2j\left(\prod_{i=1}^{m-j}(1 - \cos(x_i\pi/2))\right)(1 - \text{sen}(x_{m-j+1}\pi/2))$$

$$f_m(\vec{x}) = x_m + 2m(1 - x_1 \cos^2(5x_1\pi))$$

Donde:

$$x_{i=1:m-1} = r\_sum(\{y_{(i-1)k/(m-1)+1}, ..., y_{ik/(m-1)}\}, \{1, ..., 1\})$$
$$x_m = r\_sum(\{y_{k+1}, ..., y_{k+l/2}\}, \{1, ..., 1\})$$
$$y_{i=1:k} = y'_i$$
$$y_{i=k+1:k+l/2} = r\_nonsep(\{y'_{k+2(i-k)-1}, y'_{k+2(i-k)}\}, 2)$$
$$y'_{i=1:k} = z_i/(2i)$$
$$y'_{i=k+1:n} = s\_linear(z_i/(2i), 0.35)$$

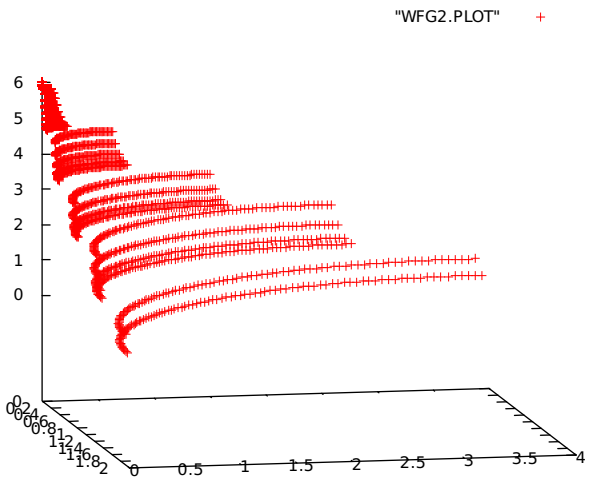

Figura A.14: Frente de Pareto verdadero de WFG2 en tres dimensiones

El problema de prueba WFG3 es no separable pero es unimodal. Su frente de Pareto verdadero es lineal (ver figura A.15), y se define de la siguiente forma:

Dado:

$$\vec{z} = \{z_1, ..., z_k, z_{k+1}, ..., z_n\}$$

Minimizar:

$$f_1(\vec{x}) = x_m + 2\prod_{i=1}^{m-1} x_i$$
$$f_{j=2:m-1}(\vec{x}) = x_m + 2j\left(\prod_{i=1}^{m-j} x_i\right)(1 - x_{m-j+1})$$
$$f_m(\vec{x}) = x_m + 2m(1 - x_1)$$

Donde:

$$x_{i=1} = u_i$$
$$x_{i=2:m-1} = x_m(u_i - 0.5) + 0.5$$
$$x_m = r\_sum(\{y_{k+1}, ..., y_{k+l/2}\}, \{1, ..., 1\})$$
$$u_i = r\_sum(\{y_{(i-1)k/(m-1)+1}, ..., y_{ik/(m-1)}\}, \{1, ..., 1\})$$
$$y_{i=1:k} = y'_i$$
$$y_{i=k+1:k+l/2} = r\_nonsep(\{y'_{k+2(i-k)-1}, y'_{k+2(i-k)}\}, 2)$$
$$y'_{i=1:k} = z_i/(2i)$$
$$y'_{i=k+1:n} = s\_linear(z_i/(2i), 0.35)$$

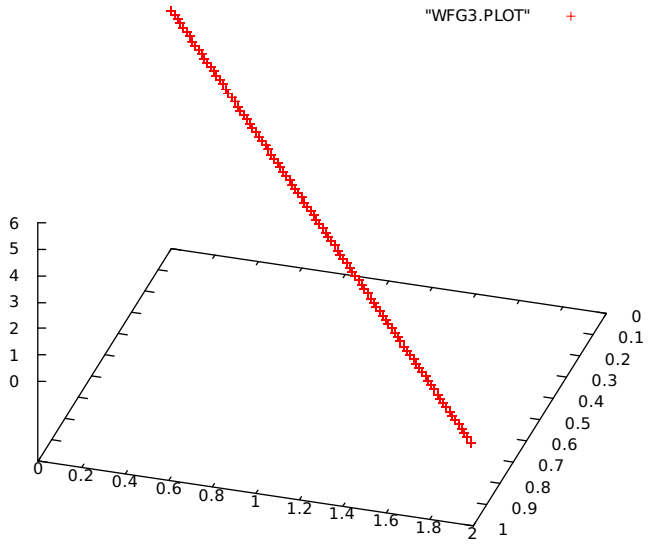

Figura A.15: Frente de Pareto verdadero de WFG3 en tres dimensiones

El problema de prueba WFG4 es separable pero altamente multimodal. Su frente de Pareto verdadero es cóncavo (ver figura A.16), y se define de la siguiente forma:

Dado:

$$\vec{z} = \{z_1, ..., z_k, z_{k+1}, ..., z_n\}$$

Minimizar:

$$f_1(\vec{x}) = x_m + 2\prod_{i=1}^{m-1} \text{sen}(x_i\pi/2)$$
$$f_{j=2:m-1}(\vec{x}) = x_m + 2j\left(\prod_{i=1}^{m-j}(\text{sen}(x_i\pi/2))\right)\cos(x_{m-j+1}\pi/2)$$

$$f_m(\vec{x}) = x_m + 2m\cos(x_1\pi/2)$$

Donde:

$$x_{i=1:m-1} = r\_sum(\{y_{(i-1)k/(m-1)+1}, ..., y_{ik/(m-1)}\}, \{1, ..., 1\})$$
$$x_m = r\_sum(\{y_{k+1}, ..., y_n\}, \{1, ..., 1\})$$
$$y_{i=1:n} = s\_multi(z_i/(2i), 30, 10, 0.35)$$

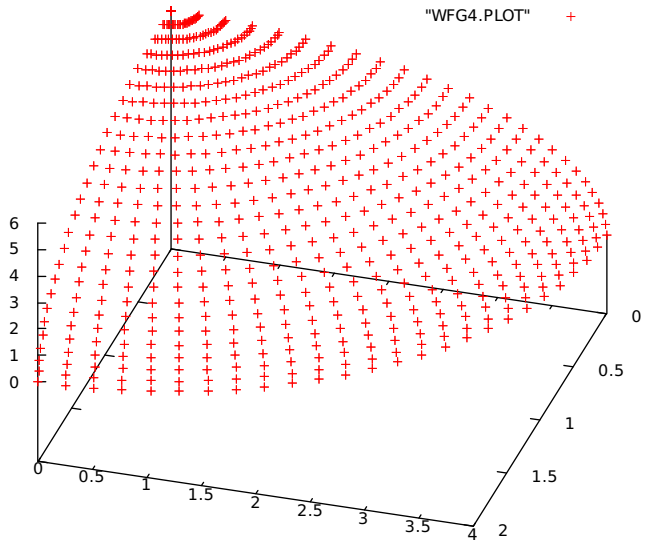

Figura A.16: Frente de Pareto verdadero de WFG4 en tres dimensiones

El problema de prueba WFG5 es separable y deceptivo. Su frente de Pareto verdadero es cóncavo(ver figura A.17), y se define de la siguiente forma:

Dado:

$$\vec{z} = \{z_1, ..., z_k, z_{k+1}, ..., z_n\}$$

Minimizar:

$$f_1(\vec{x}) = x_m + 2\prod_{i=1}^{m-1}\text{sen}(x_i\pi/2)$$
$$f_{j=2:m-1}(\vec{x}) = x_m + 2j\left(\prod_{i=1}^{m-j}(\text{sen}(x_i\pi/2))\right)\cos(x_{m-j+1}\pi/2)$$
$$f_m(\vec{x}) = x_m + 2m\cos(x_1\pi/2)$$

Donde:

$$x_{i=1:m-1} = r\_sum(\{y_{(i-1)k/(m-1)+1}, ..., y_{ik/(m-1)}\}, \{1, ..., 1\})$$

$$x_m = r\_sum(\{y_{k+1}, ..., y_n\}, \{1, ..., 1\})$$
$$y_{i=1:n} = s\_decept(z_i/(2i), 0.35, 0.001, 0.05)$$

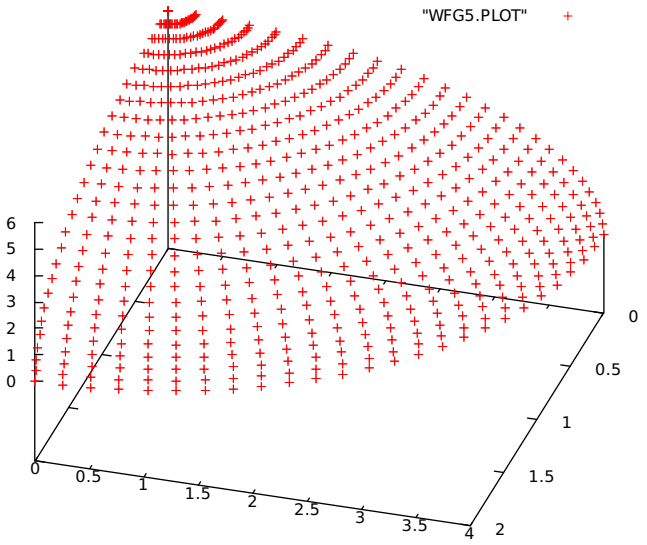

Figura A.17: Frente de Pareto verdadero de WFG5 en tres dimensiones

El problema de prueba WFG6 es no separable y unimodal. Su frente de Pareto verdadero es cóncavo (ver figura A.18), y se define de la siguiente forma:

Dado:

$$\vec{z} = \{z_1, ..., z_k, z_{k+1}, ..., z_n\}$$

Minimizar:

$$f_1(\vec{x}) = x_m + 2\prod_{i=1}^{m-1} \text{sen}(x_i\pi/2)$$
$$f_{j=2:m-1}(\vec{x}) = x_m + 2j\left(\prod_{i=1}^{m-j}(\text{sen}(x_i\pi/2))\right)\cos(x_{m-j+1}\pi/2)$$
$$f_m(\vec{x}) = x_m + 2m\cos(x_1\pi/2)$$

Donde:

$$x_{i=1:m-1} = r\_nonsep(\{y_{(i-1)k/(m-1)+1}, ..., y_{ik/(m-1)}\}, k/(m-1))$$
$$x_m = r\_nonsep(\{y_{k+1}, ..., y_n\}, l)$$
$$y_{i=1:k} = z_i/(2i)$$
$$y_{i=k+1:n} = s\_linear(z_i/(2i), 0.35)$$

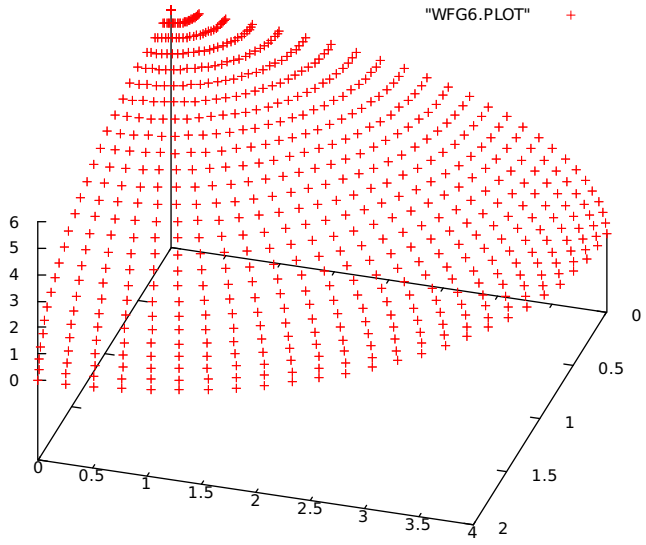

Figura A.18: Frente de Pareto verdadero de WFG6 en tres dimensiones

El problema de prueba WFG7 es separable y unimodal. Su frente de Pareto verdadero es cóncavo (ver figura A.19), y se define de la siguiente forma:

Dado:

$$\vec{z} = \{z_1, ..., z_k, z_{k+1}, ..., z_n\}$$

Minimizar:

$$f_1(\vec{x}) = x_m + 2\prod_{i=1}^{m-1} \sin(x_i\pi/2)$$
$$f_{j=2:m-1}(\vec{x}) = x_m + 2j\left(\prod_{i=1}^{m-j}(\sin(x_i\pi/2))\right)\cos(x_{m-j+1}\pi/2)$$
$$f_m(\vec{x}) = x_m + 2m\cos(x_1\pi/2)$$

Donde:

$$x_{i=1:m-1} = r\_sum(\{y_{(i-1)k/(m-1)+1}, ..., y_{ik/(m-1)}\}, \{1, ..., 1\})$$
$$x_m = r\_sum(\{y_{k+1}, ..., y_n\}, \{1, ..., 1\})$$
$$y_{i=1:k} = y'_i$$
$$y_{i=k+1:n} = s\_linear(y'_i, 0.35)$$
$$y'_{i=1:k} = b\_param(z_i/(2i), r\_sum(\{z_{i+1}/(2(i+1)), ..., z_n/(2n)\},$$
$$\{1, ..., 1\}, \tfrac{0.98}{49.98}, 0.02, 50)$$
$$y'_{i=k+1:n} = z_i/(2i)$$

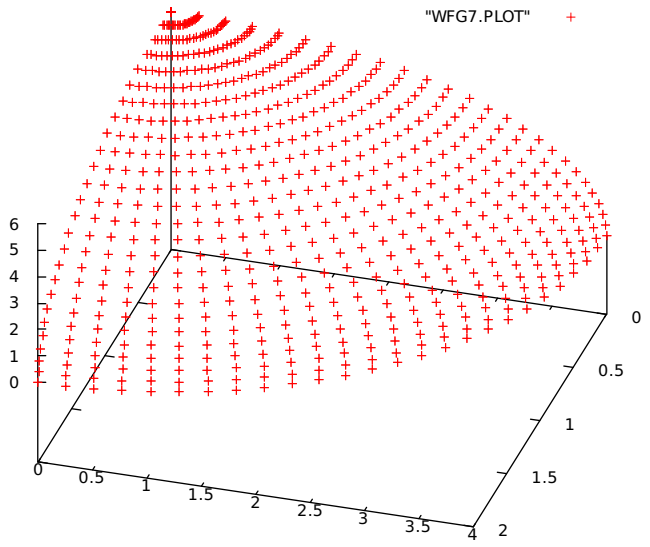

Figura A.19: Frente de Pareto verdadero de WFG7 en tres dimensiones

El problema de prueba WFG8 es no separable y unimodal. Su frente de Pareto verdadero es cóncavo (ver figura A.20), y se define de la siguiente forma:

Dado:

$$\vec{z} = \{z_1, ..., z_k, z_{k+1}, ..., z_n\}$$

Minimizar:

$$f_1(\vec{x}) = x_m + 2\prod_{i=1}^{m-1} \text{sen}(x_i\pi/2)$$
$$f_{j=2:m-1}(\vec{x}) = x_m + 2j\left(\prod_{i=1}^{m-j}(\text{sen}(x_i\pi/2))\right)\cos(x_{m-j+1}\pi/2)$$
$$f_m(\vec{x}) = x_m + 2m\cos(x_1\pi/2)$$

Donde:

$$x_{i=1:m-1} = r\_sum(\{y_{(i-1)k/(m-1)+1}, ..., y_{ik/(m-1)}\}, \{1, ..., 1\})$$
$$x_m = r\_sum(\{y_{k+1}, ..., y_n\}, \{1, ..., 1\})$$
$$y_{i=1:k} = y'_i$$
$$y_{i=k+1:n} = s\_linear(y'_i, 0.35)$$
$$y'_{i=k} = z_i/(2i)$$
$$y'_{i=k+1:n} = b\_param(z_i/(2i), r\_sum(\{z_1/2, ..., z_{i-1}/(2(i-1))\},$$
$$\{1, ..., 1\}, \tfrac{0.98}{49.98}, 0.02, 50)$$

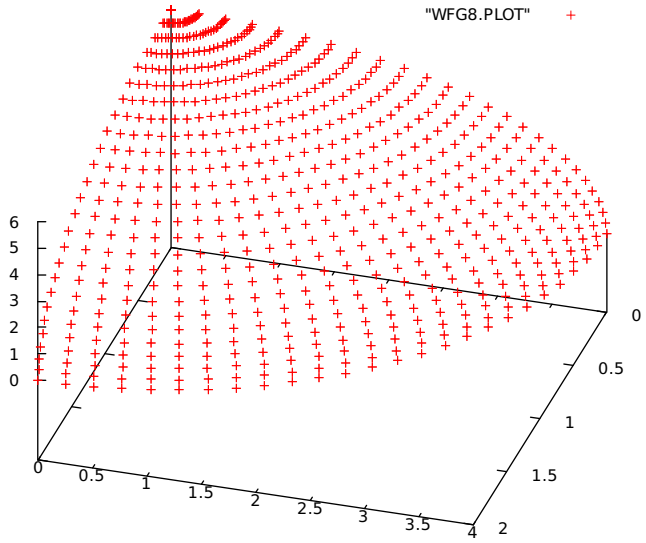

Figura A.20: Frente de Pareto verdadero de WFG8 en tres dimensiones

El problema de prueba WFG9 es no separable, multimodal y deceptivo. Su frente de Pareto verdadero es cóncavo (ver figura A.21), y se define de la siguiente forma:

Dado:

$$\vec{z} = \{z_1, ..., z_k, z_{k+1}, ..., z_n\}$$

Minimizar:

$$f_1(\vec{x}) = x_m + 2\prod_{i=1}^{m-1} \text{sen}(x_i\pi/2)$$
$$f_{j=2:m-1}(\vec{x}) = x_m + 2j\left(\prod_{i=1}^{m-j}(\text{sen}(x_i\pi/2))\right)\cos(x_{m-j+1}\pi/2)$$
$$f_m(\vec{x}) = x_m + 2m\cos(x_1\pi/2)$$

Donde:

$$x_{i=1:m-1} = r\_nonsep(\{y_{(i-1)k/(m-1)+1}, ..., y_{ik/(m-1)}\}, k/(m-1))$$
$$x_m = r\_nonsep(\{y_{k+1}, ..., y_n\}, l)$$
$$y_{i=1:k} = s\_decept(y'_i, 0.35, 0.001, 0.05)$$
$$y_{i=k+1:n} = s\_multi(y'_i, 30, 95, 0.35)$$
$$y'_{i=1:n-1} = b\_param(z_i/(2i), r\_sum(\{z_{i+1}/(2(i+1)), ..., z_n/(2n)\},$$
$$\{1, ..., 1\}, \tfrac{0.98}{49.98}, 0.02, 50)$$
$$y'_n = z_n/(2n)$$

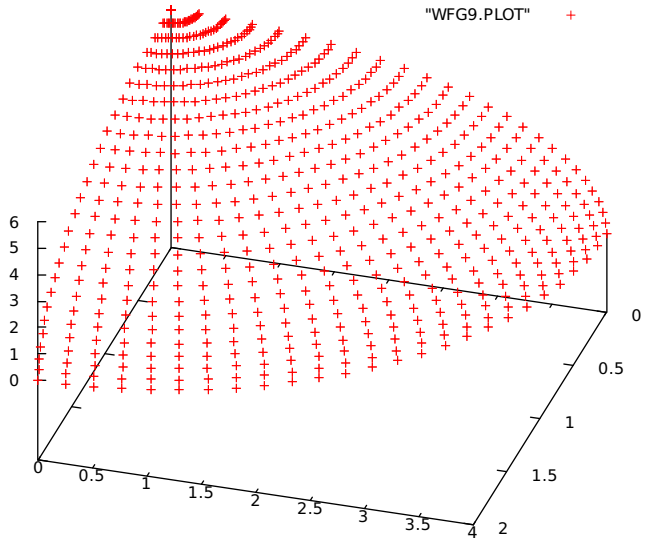

Figura A.21: Frente de Pareto verdadero de WFG9 en tres dimensiones

Apéndice B

Indicadores de desempeño para optimización multi-objetivo

A fin de poder determinar si un AEMO tiene un mejor desempeño, se requiere de indicadores de desempeño. Normalmente, buscamos medir la convergencia y la uniformidad en la distribución de las soluciones obtenidas a lo largo del frente de Pareto. En esta tesis se utilizaron tres indicadores.

1. Hipervolumen (HV)
2. Distancia Generacional Invertida Más (IGD+)
3. Espaciado (S)

Cada uno de ellos se define y describe a continuación.

B.1. Hipervolumen

El hipervolumen se obtiene calculando el volumen (en el espacio de las funciones objetivo) del conjunto de soluciones no dominadas Q que minimizan el POM. Para cada solución $i \in Q$, se genera un hipercubo v_i con un punto de referencia W y la solución i como la esquina diagonal del hipercubo. El punto de referencia W se puede generar mediante la construcción de un vector de los peores valores del vector de funciones objetivo. El hipervolumen (HV) se calcula entonces como la unión de todos los hipercubos encontrados, como se muestra a continuación:

$$I_{HV} = volumen\left(\bigcup_{i=1}^{|Q|} v_i\right) \tag{B.1}$$

Este indicador permite medir convergencia y anchura de la aproximación obtenida (es decir, la prolongación del frente hacia sus extremos). El principal inconveniente de este indicador, es que no se conoce ningún algoritmo que determine su valor en tiempo polinomial, lo cual, limita un tanto su uso en la práctica. Sin embargo, existe una gran variedad de algoritmos para determinar de forma exacta el valor de este indicador, si bien éstos requieren un elevado costo computacional.

Bringmann y Friedrich demostraron que el calcular el hipervolumen como indicador en alta dimensionalidad es considerado un problema NP-duro [104], lo cual implica que cualquier algoritmo será excesivamente costoso para calcular el hipervolumen en problemas con muchos objetivos.

B.2. IGD+

El indicador denominado *Inverted Generational Distance Plus* ($IGD+$) propuesto por Ishibuchi [105] es una variante de la Distancia Generacional Invertida (IGD) [106] el cual es débilmente compatible con la dominancia de Pareto (a diferencia de IGD que no es compatible con la dominancia de Pareto). Este indicador se calcula como una distancia promedio entre las soluciones aproximadas P' y el verdadero frente de Pareto $\tilde{P}$, siendo mejor el conjunto de soluciones que posea el menor valor de I_{IGD+}. Sin embargo, para la mayoría de los problemas de prueba no es práctico calcular el frente de Pareto en su totalidad. Debido a esto, el valor de I_{IGD+} se obtiene utilizando un conjunto $\tilde{P}$ representativo del verdadero frente de Pareto. La expresión matemática que define a I_{IGD+} es:

$$I_{IGD+}(P', \tilde{P}) = \frac{\sum_{v \in \tilde{P}} d(v, P')}{\left|\tilde{P}\right|} \tag{B.2}$$

Donde $d(v, P')$ es la mínima distancia euclidiana entre $v \in \tilde{P}$ y las soluciones en P' y se representa de la siguiente manera:

$$d(v, P') = \sqrt{\sum_{k=1}^{m} (max\{v_k - P'_k, 0\})^2} \tag{B.3}$$

Este indicador mide convergencia y distribución uniforme a lo largo del frente de Pareto.

B.3. Espaciado

Es uno de los indicadores más populares para medir la uniformidad de la distribución de las soluciones no dominadas generadas por un AEMO. Fue propuesta por Schott [22], y calcula la distancia relativa entre soluciones consecutivas. Por lo tanto, se define de la siguiente forma:

$$I_S = \sqrt{\frac{1}{|Q|}\sum_{i=1}^{|Q|}(d_i - \bar{d})^2} \tag{B.4}$$

Donde $d_i = min_{k \in Q \wedge k \neq i} \sum_{j=1}^{m} |f_j^i - f_j^k|$ y $\bar{d}$ es el valor promedio de todas las distancias y se define como: $\bar{d} = \sum_{i=1}^{|Q|} \frac{d_i}{|Q|}$. Cuando las soluciones se encuentran esparcidas uniformemente entre sí, la distancia tendrá un valor pequeño. Por lo tanto, el algoritmo que encuentre una aproximación al conjunto de óptimos de Pareto teniendo un menor valor de espaciado es mejor. El mejor valor posible para este indicador es cero.

Bibliografía

[1] M Zambrano-Bigiarini, M Clerc, and R Rojas. Standard Particle Swarm Optimisation 2011 at CEC-2013: A baseline for future PSO improvements. *2013 IEEE Congress on Evolutionary Computation, IEEE press*, pages 2337–2344, 2013.

[2] Qingfu Zhang and Hui Li. MOEA/D: A Multiobjective Evolutionary Algorithm Based on Decomposition. *IEEE Transactions on Evolutionary Computation*, 11(6):712–731, December 2007.

[3] Antonio J. Nebro, Juan J. Durillo, Jose Garcia-Nieto, Carlos A. Coello Coello, Francisco Luna, and E. Alba. SMPSO: A New PSO-based Metaheuristic for Multi-objective Optimization. In *2009 IEEE Symposium on Computational Intelligence in Multi-Criteria Decision-Making (MCDM'2009)*, pages 66–73, Nashville, TN, USA, March 30 - April 2 2009. IEEE Press. ISBN 978-1-4244-2764-2.

[4] Saúl Zapotecas Martínez and Carlos A. Coello Coello. A Multi-objective Particle Swarm Optimizer Based on Decomposition. In *2011 Genetic and Evolutionary Computation Conference (GECCO'2011)*, pages 69–76, Dublin, Ireland, July 12-16 2011. ACM Press.

[5] J Kennedy and R Eberhart. Particle swarm optimization. *IEEE International Conference on Neural Networks*, pages 1942–1948, vol. 4, 1995.

[6] M Clerc. Standard Particle Swarm Optimisation From 2006 to 2011. *Particle Swarm Central 2012-09-23 version*, 2012, URL: http://clerc.maurice.free.fr/pso/SPSO_descriptions.pdf.

[7] Thomas Bäck. Evolutionary algorithms in theory and practice: evolution strategies, evolutionary programming, genetic algorithms. *Oxford University Press*, 1996.

[8] John Henry Holland. Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence. *Bradford Books, MIT Press*, 1992.

[9] Zbigniew Michalewicz Thomas Bäck, David B. Fogel. Handbook of Evolutionary Computation. *Institute of Physics Publishing and Oxford University Press*, 1997.

[10] Ingo Rechenberg. Evolutionsstrategie: Optimierung technischer Systeme nach Prinzipien der biologischen Evolution. *Frommann-Holzboog, Stuttgart, Alemania*, 1973.

[11] Schwefel Hans Paul. Numerical Optimization of Computer Models. *John Wiley & Sons Ltd, New York, USA*, 1981.

[12] David B. Fogel. Evolutionary Computation. Toward a New Philosophy of Machine Intelligence. *The Institute of Electrical and Electronic Enginers, New York, NY, USA*, 1998.

[13] Carlos M Fonseca and Peter J Fleming. Multiobjective Optimization. In Thomas Bäck, David B Fogel, and Zbigniew Michalewicz, editors, *Handbook of Evolutionary Computation*, volume 1, pages C4.5:1—-C4.5:9. Institute of Physics Publishing and Oxford University Press, 1997.

[14] Carlos A Coello Coello, Gary B Lamont, and David A Van Veldhuizen. *Evolutionary Algorithms for Solving Multi-Objective Problems*. Springer, New York, second edition, 2007. ISBN 978-0-387-33254-3.

[15] R. C. Eberhart, R Dobbins, and P. K. Simpson. Computational Intelligence PC Tools. *Morgan Kaufmann Publishers*, 1996.

[16] J Kennedy and R. C. Eberhart. Swarm Intelligence. *Morgan Kaufmann Publishers, San Francisco California*, 2001.

[17] A. P. Engelbrecht. Computational Intelligence: An Introduction. *JohnWiley & Sons, England*, 2002.

[18] M Clerc. *Particle Swarm Optimization*. ISTE(International Scientific and Technical Encyclopedia), 2006.

[19] Ivan Chaman García, Carlos A Coello Coello, and Alfredo Arias-Montaño. MOPSOhv: A New Hypervolume-based Multi-Objective Particle Swarm Optimizer. In *2014 IEEE Congress on Evolutionary Computation (CEC'2014)*, pages 266–273, Beijing, China, 2014. IEEE Press.

[20] Fei Li, Jianchang Liu, Shubin Tan, and Xia Yu. R2-MOPSO: A multi-objective particle swarm optimizer based on R2-indicator and decomposition. In *2015 IEEE Congress on Evolutionary Computation (CEC)*, pages 3148–3155. IEEE, may 2015.

[21] Wei Peng and Qingfu Zhang. A Decomposition-based Multi-objective Particle Swarm Optimization Algorithm for Continuous Optimization Problems. In T Y Lin, X Hu, Q Liu, X Shen, J Xia, and J Wang, editors, *2008 IEEE International Conference on Granular Computing*, pages 534–537. IEEE Press, Hangzhou, China, 2008.

[22] Kalyanmoy Deb. *Multi-Objective Optimization using Evolutionary Algorithms*. John Wiley & Sons, Chichester, UK, 2001. ISBN 0-471-87339-X.

[23] Dimo Brockhoff, Tobias Friedrich, and Hebbinghaus. Failure of Pareto-based MOEAs: Does Non-dominated Really Mean Near to Optimal? In *Proceedings of the Congress on Evolutionary Computation 2001 (CEC'2001)*, volume 2, pages 957–962, Piscataway, New Jersey, may 2001. IEEE Service Center.

[24] Robin C Purshouse and Peter J Fleming. Evolutionary Multi-Objective Optimisation: An Exploratory Analysis. In *Proceedings of the 2003 Congress on Evolutionary Computation (CEC'2003)*, volume 3, pages 2066–2073, Canberra, Australia, 2003. IEEE Press.

[25] Bingdong Li, Jinlong Li, Ke Tang, and Xin Yao. Many-Objective Evolutionary Algorithms: A Survey. *ACM Computing Surveys*, 48(1), 2015.

[26] Carlos M Fonseca and Peter J Fleming. Multiobjective Optimization and Multiple Constraint Handling with Evolutionary Algorithms—Part I: A Unified Formulation. *IEEE Transactions on Systems, Man, and Cybernetics, Part A: Systems and Humans*, 28(1):26–37, 1998.

[27] Robin C Purshouse and Peter J Fleming. On the Evolutionary Optimization of Many Conflicting Objectives. *IEEE Transactions on Evolutionary Algorithms*, 11(6):770–784, 2007.

[28] Joshua Knowles and David Corne. Quantifying the Effects of Objective Space Dimension in Evolutionary Multiobjective Optimization. In Shigeru Obayashi, Kalyanmoy Deb, Carlo Poloni, Tomoyuki Hiroyasu, and Tadahiko Murata, editors, *Evolutionary Multi-Criterion Optimization, 4th International Conference, EMO 2007*, pages 757–771, Matshushima, Japan, 2007. Springer. Lecture Notes in Computer Science Vol. 4403.

[29] Hisao Ishibuchi, Naoya Akedo, and Yusuke Nojima. Behavior of Multiobjective Evolutionary Algorithms on Many-Objective Knapsack Problems. *IEEE Transactions on Evolutionary Computation*, 19(2):264–283, 2015.

[30] Yaochu Jin and Bernhard Sendhoff. Connectedness, Regularity and the Success of Local Search in Evolutionary Multi-objective Optimization. In *Proceedings of the 2003 Congress on Evolutionary Computation (CEC'2003)*, volume 3, pages 1910–1917, Canberra, Australia, 2003. IEEE Press.

[31] David J Walker, Richard M Everson, and Jonathan E Fieldsend. Visualizing Mutually Nondominating Solution Sets in Many-Objective Optimization. *IEEE Transactions on Evolutionary Computation*, 17(2):165–184, 2013.

[32] J. L. Bentley, H. T. Kung, M. Schkolnick and C. D. Thompson. On the average number of maxima in a set of vectors and applications. *ACM, 25(4)*, pages 536–543, 1978.

[33] M Farina and P Amato. On the Optimal Solution Definition for Many-criteria Optimization Problems. pages 233–238, 2002.

[34] Olivier Teytaud. On the hardness of offline multi-objective optimization. *Evolutionary Computation*, 15(4):475–491, December 2007.

[35] Dimo Brockhoff, Tobias Friedrich, Nils Hebbinghaus, Christian Klein, Frank Neumann, and Eckart Zitzler. Do Additional Objectives Make a Problem Harder? In Dirk Thierens, editor, *2007 Genetic and Evolutionary Computation Conference (GECCO'2007)*, volume 1, pages 765–772, London, UK, 2007. ACM Press.

[36] Oliver Schütze, Adriana Lara, and Carlos A Coello Coello. On the Influence of the Number of Objectives on the Hardness of a Multiobjective Optimization Problem. *IEEE Transactions on Evolutionary Computation*, 15(4):444–455, 2011.

[37] Marco Laumanns, Lothar Thiele, Kalyanmoy Deb, and Eckart Zitzler. Combining Convergence and Diversity in Evolutionary Multi-objective Optimization. *Evolutionary Computation*, 10(3):263–282, 2002.

[38] Kalyanmoy Deb, Manikanth Mohan, and Shikhar Mishra. Evaluating the ϵ-Domination Based Multi-Objective Evolutionary Algorithm for a Quick Computation of Pareto-Optimal Solutions. *Evolutionary Computation*, 13(4):501–525, 2005.

[39] Hiroyuki Sato, Hernán E Aguirre, and Kiyoshi Tanaka. Controlling Dominance Area of Solutions and Its Impact on the Performance of MOEAs. In Shigeru Obayashi, Kalyanmoy Deb, Carlo Poloni, Tomoyuki Hiroyasu, and Tadahiko Murata, editors, *Evolutionary Multi-Criterion Optimization, 4th International Conference, EMO 2007*, pages 5–20, Matshushima, Japan, 2007. Springer. Lecture Notes in Computer Science Vol. 4403.

[40] Ikeda Kokolo, Kita Hajime, and Kobayashi Shigenobu. Failure of Pareto-based MOEAs: Does Non-dominated Really Mean Near to Optimal? In *Proceedings of the Congress on Evolutionary Computation 2001 (CEC'2001)*, volume 2, pages 957–962, Piscataway, New Jersey, may 2001. IEEE Service Center.

[41] M Farina and P Amato. On the Optimal Solution Definition for Many-criteria Optimization Problems. In *Proceedings of the NAFIPS-FLINT International Conference'2002*, pages 233–238, Piscataway, New Jersey, 2002. IEEE Service Center.

[42] Nicole Drechsler, Rolf Drechsler, and Bernd Becker. Multi-objective Optimisation Based on Relation Favour. In Eckart Zitzler, Kalyanmoy Deb, Lothar Thiele, Carlos A Coello Coello, and David Corne, editors, *First International Conference on*

Evolutionary Multi-Criterion Optimization, pages 154–166. Springer-Verlag. Lecture Notes in Computer Science No. 1993, 2001.

[43] Xiufen Zou, Yu Chen, Minzhong Liu, and Lishan Kang. A new evolutionary algorithm for solving many-objective optimization problems. *IEEE transactions on systems, man, and cybernetics. Part B, Cybernetics : a publication of the IEEE Systems, Man, and Cybernetics Society*, 38(5):1402–1412, 2008.

[44] S. F. Adra Fleming and P. J. Diversity management in evolutionary many-objective optimization. *IEEE Transactions on Evolutionary Computation 15, 2*, pages 183–195, 2011.

[45] M.Li, S.Yang and X.Liu. Shift-based density estimation for Pareto-based algorithms in many-objective optimization. *IEEE Transactions on Evolutionary Computation*, 18(3):348–365, 2014.

[46] Tapabrata Ray, Md Asafuddoula, and Amitay Isaacs. A Steady State Decomposition Based Quantum Genetic Algorithm for Many Objective Optimization. In *2013 IEEE Congress on Evolutionary Computation (CEC'2013)*, pages 2817–2824, Cancún, México, 2013. IEEE Press. ISBN 978-1-4799-0454-9.

[47] Ioannis Giagkiozis, Robin C Purshouse, and Peter J Fleming. Generalized Decomposition. In Robin C Purshouse, Peter J Fleming, Carlos M Fonseca, Salvatore Greco, and Jane Shaw, editors, *Evolutionary Multi-Criterion Optimization, 7th International Conference, EMO 2013*, pages 428–442. Springer. Lecture Notes in Computer Science Vol. 7811, Sheffield, UK, 2013.

[48] Mario Garza Fabre, Gregorio Toscano Pulido, and Carlos A Coello Coello. Ranking Methods for Many-Objective Problems. In Arturo Hernández Aguirre, Raúl Monroy Borja, and Carlos Alberto Reyes García, editors, *MICAI 2009: Advances in Artificial Intelligence. 8th Mexican International Conference on Artificial Intelligence*, pages 633–645, Guanajuato, México, 2009. Springer. Lecture Notes in Artificial Intelligence Vol. 5845.

[49] P. Bentley Wakefield and J. Finding acceptable solutions in the Pareto-optimal range using multi-objective genetic algorithms. *In Soft Computing in Engineering Design and Manufacturing. Springer*, pages 231–240, 1998.

[50] Richard Balling and Scott Wilson. The Maximin Fitness Function for Multi-objective Evolutionary Computation: Application to City Planning. In Lee Spector, Erik D Goodman, Annie Wu, W B Langdon, Hans-Michael Voigt, Mitsuo Gen, Sandip Sen, Marco Dorigo, Shahram Pezeshk, Max H Garzon, and Edmund Burke, editors, *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'2001)*, pages 1079–1084, San Francisco, California, 2001. Morgan Kaufmann Publishers.

[51] Adriana Menchaca-Mendez and Carlos A. Coello Coello. Selection Operators Based on Maximin Fitness Function for Multi-Objective Evolutionary Algorithms. pages 215–229, March 19-22 2013.

[52] Johannes Bader and Eckart Zitzler. HypE: An Algorithm for Fast Hypervolume-Based Many-Objective Optimization. *Evolutionary Computation*, 19(1):45–76, Spring, 2011.

[53] Michael Emmerich, Nicola Beume, and Boris Naujoks. An EMO Algorithm Using the Hypervolume Measure as Selection Criterion. In Carlos A Coello Coello, Arturo Hernández Aguirre, and Eckart Zitzler, editors, *Evolutionary Multi-Criterion Optimization. Third International Conference, EMO 2005*, pages 62–76, Guanajuato, México, 2005. Springer. Lecture Notes in Computer Science Vol. 3410.

[54] P. A. N. Bosman. On gradients and hybrid evolutionary algorithms for real-valued multiobjective optimization. *IEEE Transactions on Evolutionary Computation 16, 1*, pages 51–69, 2012.

[55] N. Beume Rudolph and G. Faster S-metric calculation by considering dominated hypervolume as Klee's measure problem. *In Proceedings of the 2nd IASTED Conference on Computational Intelligence (CI'06)*, pages 231–236, 2006.

[56] Adriana Menchaca-Mendez and Carlos A. Coello Coello. Selection mechanisms based on the maximin fitness function to solve multi-objective optimization problems. *Information Sciences*, 332:131–152, March 1 2016.

[57] Karl Bringmann, Tobias Friedrich, Frank Neumann, and Markus Wagner. Approximation-Guided Evolutionary Multi-Objective Optimization. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence (IJCAI 2011)*, pages 1198–1203, Barcelona, Spain, 2011. AAAI Press.

[58] Roman Denysiuk, Lino Costa, and Isabel Espírito Santo. Many-Objective Optimization using Differential Evolution with Variable-Wise Mutation Restriction. In *2013 Genetic and Evolutionary Computation Conference (GECCO'2013)*, pages 591–598, New York, USA, 2013. ACM Press. ISBN 978-1-4503-1963-8.

[59] Eckart Zitzler and Simon Künzli. Indicator-based Selection in Multiobjective Search. In Xin Yao et al., editor, *Parallel Problem Solving from Nature - PPSN VIII*, pages 832–842, Birmingham, UK, September 2004. Springer-Verlag. Lecture Notes in Computer Science Vol. 3242.

[60] Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and T Meyarivan. A Fast and Elitist Multiobjective Genetic Algorithm: NSGA–II. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, 2002.

[61] Nicola Beume, Boris Naujoks, and Michael Emmerich. SMS-EMOA: Multiobjective selection based on dominated hypervolume. *European Journal of Operational Research*, 181(3):1653–1669, 2007.

[62] Eckart Zitzler, Marco Laumanns, and Lothar Thiele. SPEA2: Improving the Strength Pareto Evolutionary Algorithm. Technical Report 103, Computer Engineering and Networks Laboratory (TIK), Swiss Federal Institute of Technology (ETH) Zurich, Gloriastrasse 35, CH-8092 Zurich, Switzerland, may 2001.

[63] Alan Díaz-Manríquez, Gregorio Toscano-Pulido, Carlos A Coello Coello, and Ricardo Landa-Becerra. A Ranking Method Based on the R2 Indicator for Many-Objective Optimization. In *2013 IEEE Congress on Evolutionary Computation (CEC'2013)*, pages 1523–1530, Cancún, México, 2013. IEEE Press.

[64] David E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Publishing Company, Reading, Massachusetts, 1989.

[65] Raquel Hernández Gómez and Carlos A Coello Coello. MOMBI: A New Metaheuristic for Many-Objective Optimization Based on the R2 Indicator. In *2013 IEEE Congress on Evolutionary Computation (CEC'2013)*, pages 2488–2495, Cancún, México, 2013. IEEE Press.

[66] Raquel Hernández Gómez and Carlos A. Coello Coello. Improved Metaheuristic Based on the $R2$ Indicator for Many-Objective Optimization. In *2015 Genetic and Evolutionary Computation Conference (GECCO 2015)*, pages 679–686, Madrid, Spain, July 11-15 2015. ACM Press. ISBN 978-1-4503-3472-3.

[67] Dũng H. Phan and Junichi Suzuki. R2-IBEA: R2 Indicator Based Evolutionary Algorithm for Multiobjective Optimization. In *2013 IEEE Congress on Evolutionary Computation (CEC'2013)*, pages 1836–1845, Cancún, México, 20-23 June 2013. IEEE Press. ISBN 978-1-4799-0454-9.

[68] Joshua Knowles and David Corne. On Metrics for Comparing Nondominated Sets. In *Congress on Evolutionary Computation (CEC'2002)*, volume 1, pages 711–716, Piscataway, New Jersey, May 2002. IEEE Service Center.

[69] Kata Praditwong and Xin Yao. A new multi-objective evolutionary optimisation algorithm: The two-archive algorithm. In Yuping Wang, Yiu-ming Cheung, and Hailin Liu, editors, *Computational Intelligence and Security, International Conference, CIS 2006*, pages 95–104, Guangzhou, China, 2007. Springer. Lecture Notes in Computer Science 4456.

[70] Bingdong Li, Jinlong Li, Ke Tang, and Xin Yao. An Improved Two Archive Algorithm for Many-Objective Optimization. In *2014 IEEE Congress on Evolutionary Computation (CEC'2014)*, pages 2869–2876, Beijing, China, 2014. IEEE Press. ISBN 978-1-4799-1488-3.

[71] H. Jain Deb and K. An evolutionary many-objective optimization algorithm using reference-point based nondominated sorting approach, part II: Handling constraints and extending to an adaptive approach. *IEEE Transactions on Evolutionary Computation 18, 4*, pages 602–622, 2014.

[72] Hans J F Moen, Nikolai B Hansen, Harald Hovland, and Jim Tørresen. Many-Objective Optimization Using Taxi-Cab Surface Evolutionary Algorithm. In Robin C Purshouse, Peter J Fleming, Carlos M Fonseca, Salvatore Greco, and Jane Shaw, editors, *Evolutionary Multi-Criterion Optimization, 7th International Conference, EMO 2013*, pages 128–142. Springer. Lecture Notes in Computer Science Vol. 7811, Sheffield, UK, 2013.

[73] Hisao Ishibuchi, Noritaka Tsukamoto, and Yusuke Nojima. Evolutionary many-objective optimization: A short review. In *2008 Congress on Evolutionary Computation (CEC'2008)*, pages 2424–2431, Hong Kong, 2008. IEEE Service Center.

[74] L Rachmawati and D Srinivasan. Preference Incorporation in Multi-objective Evolutionary Algorithms: A Survey. In *2006 IEEE Congress on Evolutionary Computation (CEC'2006)*, pages 3385–3391, Vancouver, BC, Canada, 2006. IEEE.

[75] G. W. Evans. An overview of techniques for solving multiobjective mathematical programs. *Management Science 30, 11*, pages 1268–1282, 1984.

[76] Antonio López-Jaimes, Alfredo Arias-Montaño, and Carlos A Coello Coello. Preference Incorporation to Solve Many-Objective Airfoil Design Problems. In *2011 IEEE Congress on Evolutionary Computation (CEC'2011)*, pages 1605–1612, New Orleans, Louisiana, USA, 2011. IEEE Service Center.

[77] Anne Auger, Johannes Bader, Dimo Brockhoff, and Eckart Zitzler. Articulating User Preferences in Many-objective Problems by Sampling the Weighted Hypervolume. In *2009 Genetic and Evolutionary Computation Conference (GECCO'2009)*, pages 555–562, Montreal, Canada, 2009. ACM Press. ISBN 978-1-60558-325-9.

[78] Fei-yue Qiu, Yu-shi Wu, Li-ping Wang, and Bo Jiang. Bipolar preferences dominance based evolutionary algorithm for many-objective optimization. In *2012 IEEE Congress on Evolutionary Computation (CEC'2012)*, pages 2153–2160, Brisbane, Australia, 2012. IEEE Press.

[79] Jong-Hwan Kim, Ji-Hyeong Han, Ye-Hoon Kim, Seung-Hwan Choi, and Eun-Soo Kim. Preference-Based Solution Selection Algorithm for Evolutionary Multiobjective Optimization. *IEEE Transactions on Evolutionary Computation*, 16(1):20–34, 2012.

[80] Ricardo Landa, Carlos A Coello Coello, and Gregorio Toscano-Pulido. Goal-constraint: Incorporating Preferences Through an Evolutionary ϵ-constraint Based Method. In *2013 IEEE Congress on Evolutionary Computation (CEC'2013)*, pages 741–747, Cancún, México, 2013. IEEE Press.

[81] K. Deb Chaudhuri and S. I-EMO: An interactive evolutionary multi-objective optimization tool. *Pattern Recognition and Machine Intelligence. Springer*, pages 690–695, 2005.

[82] Kalyanmoy Deb and J Sundar. Reference Point Based Multi-Objective Optimization Using Evolutionary Algorithms. In Maarten Keijzer Et al., editor, *2006 Genetic and Evolutionary Computation Conference (GECCO'2006)*, volume 1, pages 635–642, Seattle, Washington, USA, 2006. ACM Press. ISBN 1-59593-186-4.

[83] Kalyanmoy Deb and Abhishek Kumar. Interactive Evolutionary Multi-Objective Optimization and Decision-Making using Reference Direction Method. In Dirk Thierens, editor, *2007 Genetic and Evolutionary Computation Conference (GECCO'2007)*, volume 1, pages 781–788, London, UK, 2007. ACM Press.

[84] Lothar Thiele, Kaisa Miettinen, Pekka J Korhonen, and Julian Molina. A Preference-Based Evolutionary Algorithm for Multi-Objective Optimization. *Evolutionary Computation*, 17(3):411–436, 2009.

[85] Dunwei Gong, Jing Sun, and Xinfang Ji. Evolutionary algorithms with preference polyhedron for interval multi-objective optimization problems. *Information Sciences*, 233:141–161, 2013.

[86] Robin C Purshouse, Cezar Jalbŭa, and Peter J Fleming. Preference-Driven Co-evolutionary Algorithms Show Promise for Many-Objective Optimisation. In Ricardo H C Takahashi, Kalyanmoy Deb, Elizabeth F Wanner, and Salvatore Grecco, editors, *Evolutionary Multi-Criterion Optimization, 6th International Conference, EMO 2011*, pages 136–150, Ouro Preto, Brazil, 2011. Springer. Lecture Notes in Computer Science Vol. 6576.

[87] Rui Wang, Robin C Purshouse, and Peter J Fleming. Local Preference-Inspired Co-Evolutionary Algorithms. In *2012 Genetic and Evolutionary Computation Conference (GECCO'2012)*, pages 513–520, Philadelphia, USA, 2012. ACM Press.

[88] Rui Wang, Robin C Purshouse, and Peter J Fleming. On Finding Well-Spread Pareto Optimal Solutions by Preference-inspired Co-evolutionary Algorithm. In *2013 Genetic and Evolutionary Computation Conference (GECCO'2013)*, pages 695–702, New York, USA, 2013. ACM Press.

[89] Rui Wang, Robin C Purshouse, and Peter J Fleming. Preference-Inspired Coevolutionary Algorithms for Many-Objective Optimization. *IEEE Transactions on Evolutionary Computation*, 17(4):474–494, 2013.

[90] Dhish Kumar Saxena, João A Duro, Ashutosh Tiwari, Kalyanmoy Deb, and Qingfu Zhang. Objective Reduction in Many-Objective Optimization: Linear and Nonlinear Algorithms. *IEEE Transactions on Evolutionary Computation*, 17(1):77–99, 2013.

[91] D. Brockhoff Zitzler and E. Automated aggregation and omission of objectives for tackling many-objective problems. *New Developments in Multiple Objective and Goal Programming. Springer*, pages 81– 102, 2010.

[92] Antonio López Jaimes, Carlos A Coello Coello, and Debrup Chakraborty. Objective Reduction Using a Feature Selection Technique. In *2008 Genetic and Evolutionary Computation Conference (GECCO'2008)*, pages 674–680, Atlanta, USA, 2008. ACM Press.

[93] P. Mitra, C. A. Murthy and S. K. Pal. Unsupervised feature selection using feature similarity. *IEEE Transactions on Pattern Analysis and Machine Intelligence 24, 3*, pages 301–312, 2002.

[94] Bingdong Li, Jinlong Li, Ke Tang, and Xin Yao. Many-Objective Evolutionary Algorithms: A Survey. *ACM Computing Surveys*, 48(1), 2015.

[95] Dhish Kumar Saxena and Kalyanmoy Deb. Non-linear Dimensionality Reduction Procedures for Certain Large-Dimensional Multi-objective Optimization Problems: Employing Correntropy and a Novel Maximum Variance Unfolding. In Shigeru Obayashi, Kalyanmoy Deb, Carlo Poloni, Tomoyuki Hiroyasu, and Tadahiko Murata, editors, *Evolutionary Multi-Criterion Optimization, 4th International Conference, EMO 2007*, pages 772–787, Matshushima, Japan, 2007. Springer. Lecture Notes in Computer Science Vol. 4403.

[96] Dimo Brockhoff, Dhish Kumar Saxena, Kalyanmoy Deb, and Eckart Zitzler. On Handling a Large Number of Objectives A Posteriori and During Optimization. In Joshua Knowles, David Corne, and Kalyanmoy Deb, editors, *Multi-Objective Problem Solving from Nature: From Concepts to Applications*, pages 377–403. Springer, Berlin, 2008.

[97] Yuan Yuan, Hua Xu, Bo Wang, Bo Zhang, and Xin Yao. Balancing Convergence and Diversity in Decomposition-Based Many-Objective Optimizers. *IEEE Transactions on Evolutionary Computation*, 20(2):180–198, April 2016.

[98] Antonio J Nebro, Juan J Durillo, and Carlos A Coello Coello. Analysis of Leader Selection Strategies in a Multi-Objective Particle Swarm Optimizer. In *2013 IEEE Congress on Evolutionary Computation (CEC'2013)*, pages 3153–3160, Cancún, México, 2013. IEEE Press. ISBN 978-1-4799-0454-9.

[99] Kalyanmoy Deb. *Multi-Objective Optimization using Evolutionary Algorithms*. John Wiley & Sons, Chichester, UK, 2001. ISBN 0-471-87339-X.

[100] Carlos A. Coello Coello. The EMOO repository: a resource for doing research in evolutionary multiobjective optimization. *IEEE Computational Intelligence Magazine*, 1(1):37–45, February 2006.

[101] Eckart Zitzler, Kalyanmoy Deb, and Lothar Thiele. Comparison of Multiobjective Evolutionary Algorithms: Empirical Results. *Evolutionary Computation*, 8(2):173–195, 2000.

[102] Kalyanmoy Deb, Lothar Thiele, Marco Laumanns, and Eckart Zitzler. Scalable Multi-Objective Optimization Test Problems. In *Congress on Evolutionary Computation (CEC'2002)*, volume 1, pages 825–830, Piscataway, New Jersey, may 2002. IEEE Service Center.

[103] Simon Huband, Phil Hingston, Luigi Barone, and Lyndon While. A Review of Multiobjective Test Problems and a Scalable Test Problem Toolkit. *IEEE Transactions on Evolutionary Computation*, 10(5):477–506, 2006.

[104] Karl Bringmann and Tobias Friedrich. Approximating the volume of unions and intersections of high-dimensional geometric objects. *Computational Geometry-Theory and Applications*, 43(6-7):601–610, 2010.

[105] Hisao Ishibuchi, Hiroyuki Masuda, and Yusuke Nojima. A Study on Performance Evaluation Ability of a Modified Inverted Generational Distance Indicator. *Proceedings of the 2015 on Genetic and Evolutionary Computation Conference - GECCO '15*, pages 695–702, 2015.

[106] Carlos A. Coello Coello. Recent Trends in Evolutionary Multiobjective Optimization. In Ajith Abraham, Lakhmi Jain, and Robert Goldberg, editors, *Evolutionary Multiobjective Optimization: Theoretical Advances And Applications*, pages 7–32. Springer-Verlag, London, 2005. ISBN 1-85233-787-7.