Centro de Investigación y de Estudios Avanzados
del Instituto Politécnico Nacional

Unidad Zacatenco

Departamento de Computación

Uso de un Algoritmo de Cúmulo de Partículas Basado en Hipervolumen para Resolver Problemas Multi-Objetivo

Tesis que presenta

Iván Christhofer Chaman García

para obtener el Grado de

Maestro en Ciencias

en Computación

Director de la Tesis

Dr. Carlos Artemio Coello Coello

Mexico, D.F. **Octubre de 2012**

Resumen

Muchos problemas de la vida real requieren de la optimización de dos o más objetivos a la vez, los cuales, están en conflicto unos con otros al mismo tiempo. Por lo general, los problemas de optimización multi-objetivo no tienen una solución única sino un conjunto de soluciones que representan los diferentes compromisos entre los objetivos.

La mayoría de los algoritmos evolutivos multi-objetivo utilizan la dominancia de Pareto como criterio de selección. Si bien este mecanismo es efectivo en problemas con dos o tres objetivos, no es escalable ya que la proporción de soluciones incomparables que se generan crece rápidamente al aumentar el número de objetivos.

En este trabajo de tesis se utiliza la optimización mediante cúmulos de partículas (PSO), la cual es una técnica evolutiva inspirada en el comportamiento social de los bancos de peces. Se plantea una forma de hacer que un algoritmo de cúmulo de partículas resuelva problemas de optimización multi-objetivo. Al optimizador propuesto se le integra como mecanismo de selección la métrica denominada hipervolumen, a fin de obtener a los mejores líderes o guías en su proceso de búsqueda. Nuestra propuesta obtiene buenos resultados para problemas con dos y tres objetivos y presenta resultados prometedores para problemas con muchos objetivos (cuatro o más).

Abstract

Many real-life problems require optimization the simultaneous of two or more conflicting objectives. Usually, multiobjective optimization problems do not have a single solution but a set of them, representing the trade-offs among the objetives.

Most multiobjective evolutionary algorithms use Pareto dominance as their selection criterion. This mechanism is effective in problems having two or three objectives but it does not properly scale since number of incomparable solutions that this selection criterion generates rapidly grows as the number of objectives increases.

In this thesis we use Particle Swarm Optimization (PSO), which is an evolutionary technique inspired in the social behavior of fish schools. We present here a way to adapt PSOs that it can solve multiobjetive optimization problems. The proposed approach adopts as its selection mechanism, a performance measure called hypervolume to find the best leaders during the search process. Our proposed approach was able to obtain good results for problems with two and three objectives and presented promising results for problems with many objectives (more than three).

Agradecimientos

Agradezco a los excelentes profesores que hacen posible el conocimiento dentro de las aulas.

A mis compañeros de generación y a todas aquellas personas que conocí en las aulas de clases, en las canchas deportivas y en los diversos departamentos de esta institución, por todos los buenos y malos momentos que viví con ellos. A todos los que alguna vez han compartido su tiempo.

Agradezco a el CONACYT por el apoyo económico brindado durante mis estudios de maestría.

Agradezco al CINVESTAV, esta intitución de enorme calidad, que me brindó todo el apoyo durante mi estancia.

Quiero agradecerle a mi asesor de tesis, el Dr. Carlos A. Coello Coello por sus conocimientos invaluables que me brindo para llevar a cabo esta investigación, y por su gran paciencia para esperar a que este trabajo pudiera llegar a su fin.

Agradezco a los miembros del jurado, el Dr. Eduardo Arturo Rodríguez Tello y al Dr. Alfredo Arias, por sus contribuciones que hicieron al trabajo final y por el tiempo que dedicaron para revisarlo.

Agradezco a Sofia Reza, por toda su comprensión, nobleza y dedicación que en todo momento muestra hacia todos.

Agradezco a toda mi familia y amigos por su cariño, paciencia, comprensión, apoyo y nunca dejar de creer en mí.

Este trabajo de tesis se derivó del proyecto CONACyT titulado “Escalabilidad y Nuevos Esquemas Híbridos en Optimización Evolutiva Multiobjetivo” (Ref. 103570), cuyo responsable es el Dr. Carlos A. Coello Coello.

Dedicatoria

A Dios por haberme permitido llegar hasta este punto y haberme dado salud para lograr mis objetivos, además de su infinita bondad y amor.

Dedico esta tesis a todas las personas que forman y formaron parte de mi vida. Pero muy en especial a mi madre Blanca García Rendon, mi madrina Maria de los Angeles Ordoñez Robles y mi hermano Aldo C. Chaman García por enseñarme a ser la persona que soy y a quienes admiro por su fortaleza de carácter a pesar de todos los problemas a los que se han enfrentado en su vida y por permitirme llevar a cabo las metas fijadas hasta el momento e impusarme para lograrlas.

A mi padre Jeronimo Chaman Chaman que esta en presencia del Altísimo, el dedicarle este trabajo muestra el permanente apoyo que me ofreció con su espíritu alentador, contribuyendo incondicionalmente a lograr mis metas y objetivos propuestos y que al brindarme con su ejemplo a ser perseverante y darme la fuerza que me impulsó a conseguirlo.

Índice general

Índice de figuras

Índice de tablas

Introducción

Se muestra una breve introducción a este trabajo de tesis, así como el planteamiento del problema a resolverse y los objetivos planteados. En la parte final se proporciona una breve descripción del contenido de cada capítulo de este trabajo de tesis.

Antecedentes

Los algoritmos evolutivos pueden verse como una analogía del mecanismo de selección natural, cuyo principal objetivo es simular el proceso evolutivo en una computadora y usarlo para resolver problemas de optimización. El uso de algoritmos evolutivos se ha extendido a un gran número de dominios debido, sobre todo, a su simplicidad conceptual y facilidad de uso.

Un problema de optimización multi-objetivo difiere de un problema de optimización mono-objetivo en la cantidad de objetivos a optimizarse. Mientras que en un problema de optimización mono-objetivo se busca la mejor solución posible en todo el espacio de búsqueda, en un problema multi-objetivo, se tienen varios objetivos (posiblemente en conflicto), y normalmente no hay una solución única sino un conjunto de soluciones compromiso. Por lo tanto, en este último caso se requiere elegir una solución entre varias, a partir de las preferencias del usuario [Miettinen, 1999].

La principal motivación para usar algoritmos evolutivos para resolver problemas de optimización multi-objetivo es explotar su población, de manera que podamos generar varios miembros del conjunto de óptimos de Pareto en una sola ejecución del algoritmo, en lugar de tener que realizar una serie de ejecuciones por separado como en el caso de las técnicas de programación matemática. Además, los algoritmos evolutivos son menos susceptibles a la forma o la continuidad del frente de Pareto, mientras que las técnias de programación matemática suelen tener dificultades para lidiar con frentes de Pareto desconectados y no convexos [Osyczka, 1985].

Planteamiento del problema

El uso de esquemas de selección basados en la optimalidad de Pareto presenta diversas limitantes, dentro de las que destaca su pobre escalabilidad cuando se aumenta la cantidad de funciones objetivo. En años recientes, se han planteado diferentes mecanismos de selección para algoritmos evolutivos multi-objetivo que no se basan en la optimalidad de Pareto. Una de las alternativas más prometedoras es el uso de indicadores de desempeño para seleccionar soluciones. De entre los posibles indicadores que pueden utilizarse, el hi-

pervolumen (o métrica $\mathcal{S}$) es, sin duda el más popular debido, sobre todo, a sus propiedades matemáticas (se ha podido demostrar que maximizar el hipervolumen conduce a lograr convergencia). En este trabajo se plantea incorporar en un algoritmo de cúmulos de partículas un mecanismo basado en el indicador de hipervolumen para seleccionar a los mejores líderes que conducirán la búsqueda. El nuevo algoritmo deberá ser competitivo con respecto a algoritmos evolutivos multi-objetivo del estado del arte.

Objetivos

Objetivo general

Desarrollar un algoritmo multi-objetivo con base en la metaheurística conocida como cúmulos de partículas, el cual utilice el hipervolumen en su mecanismo de selección y sea competitivo con otros algoritmos evolutivos multi-objetivo.

Objetivos específicos

- Estudiar las diferentes maneras de implementar el cálculo del hipervolumen (de manera exacta o aproximada).
- Analizar diferentes algoritmos basados en cúmulos de partículas para establecer las diferencias y similitudes entre ellos.
- Diseñar un mecanismo de selección que aproveche las características del hipervolumen, e incorporárselo a un algoritmo multi-objetivo basado en cúmulos de partículas.
- Comparar el desempeño de las soluciones del algoritmo desarrallado con respecto a las soluciones de otros algoritmos utilizando un conjunto diverso de problemas de la literatura especializada.

Organización

La organización de este trabajo de tesis es la siguiente:

- En el capítulo 1 se describen algunos conceptos básicos sobre algoritmos evolutivos, optimización multiobjetivo, la metaheurística de cúmulos de partículas y las caraterísticas del indicador de hipervolumen que servirán como marco teórico para el resto del trabajo de tesis. Además, se proporciona una revision del estado del arte sobre los algoritmos evolutivos multi-objetivo.
- En el capítulo 2 se describe el algoritmo de cúmulos de partículas multi-objetivo propuesto en este trabajo de tesis, cuyo mecanismo de selección está basado en la contribución al hipervolumen y adopta un operador de turbulencia.
- En el capítulo 3 se compara el algoritmo propuesto con respecto a otros algoritmos evolutivos multi-objetivo tomados de la literatura especializada utilizando los indicadores de calidad descritos en el capítulo 1.
- En el capítulo 4 se proporcionan las conclusiones y el trabajo futuro.

Capítulo 1

Conceptos Básicos

Un número considerable de problemas de la vida real requieren de la optimización de dos o más objetivos a la vez, los cuales, en general, están en conflicto unos con otros al mismo tiempo. En este capítulo se describen algunos conceptos básicos de algoritmos evolutivos, de la optimización multi-objetivo y del algoritmo de cúmulos de partículas que servirán como marco teórico para el resto de la tesis. En este capítulo se proporcionan también las principales características del indicador llamado hipervolumen, así como un breve estado del arte sobre los algoritmos evolutivos multi-objetivo más relacionados con este trabajo de tesis.

1.1. Conceptos de computación evolutiva

La computación evolutiva consiste en técnicas estocásticas de búsqueda y optimización inspiradas en el *neodarwinismo*. A estas técnicas se les conoce genéricamente como *Algoritmos Evolutivos* y han tenido mucho éxito en la solución de problemas de optimización, debido a que presentan las siguientes características [Santana Quintero and Coello Coello, 2006]:

- No requieren de conocimientos específicos sobre el problema.
- Pueden actuar como eficaces optimizadores globales, ya que son menos propensos a quedar atrapados en óptimos locales.
- Son fáciles de comprender e implementar en forma secuencial y paralela.
- Se pueden hibridizar con otras técnicas de optimización (por ejemplo, técnicas de programación matemática).

El *neodarwinismo* se compone de tres aspectos básicos:

- **La teoría de la evolución de *Darwin***. La contribución de Darwin a los conocimientos científicos acerca de la evolución se centró directamente en la identificación del mecanismo principal de la evolución, llamado *selección natural*. Esta teoría establece que las condiciones del medio favorecen o dificultan la reprodución de los individuos para adaptarse al medio y sus cambios constantes. Esta adaptabilidad que presentan los individuos hacia los cambios del medio se sostiene con la afirmación de que los descendientes tienen variaciones no aleatorias y no deterministas que son, en parte, heredables y otras variaciones que dan lugar a una mejor sobrevivencia y éxito reproductivo, haciendo que a lo largo de las generaciones se produzca el fenómeno evolutivo [Darwin, 1959].

- **La teoría de Germoplasma de *Weismann***. [Weismann, 1893] propuso la teoría de germoplasma como un mecanismo de herencia en la teoría de Darwin. Esta teoría distingue dos componentes en los individuos: el primero, son las células germinales que contienen información hereditaria que no puede ser alterada por las habilidades adquiridas durante la vida del individuo, es decir, son las células que dan origen a la descendencia; el segundo, son las células somáticas las responsables de las funciones corporales.

- **La Genética Mendeliana o Leyes de Mendel**. Las leyes de Mendel son un conjunto de reglas sobres la transmisión por herencia entre los individuos (padres a hijos). Estas leyes fueron derivadas por [Mendel et al., 1965]:

  1. **Ley de la segregación**: establece que durante la formación de los gametos cada alelo de un par se separa del otro miembro para determinar la constitución genética del gameto.
  2. **Ley de la segregación independiente**: durante la formación de los gametos la segregación de los alelos de un par es independiente de la segregación de los alelos de otro par.
  3. **Ley de la dominancia**: cuando un individuo tiene dos alelos diferentes para un rasgo (un alelo es dominante y otro recesivo), el alelo dominante se expresa mientras que el alelo recesivo se mantiene oculto.

  De acuerdo con el *neodarwinismo*, se tienen cuatro procesos estocásticos que actuán en todas las especies [Hoffman, 1989]:

  - **La reproduccón**: es la formación de nuevos individuos a partir de otros preexistentes y, por tanto, pueden aparecer individuos diferentes a los de los progenitores, es decir, es un proceso que permite la creación de nuevos individuos.
  - **La mutación**: son cambios al azar que se producen en la composición genética de un individuo. Se originan en los cromosomas, por lo que se pueden transmitir a la descendencia durante la reproducción. Consisten generalmente en que un gen sufre alguna modificación.
  - **La competencia**: es la interacción entre los individuos que pertenecen a una comunidad, debido a la disponibilidad de ciertos recursos limitados.
  - **La selección**: las combinaciones peor adaptadas al medio se eliminan, mientras que las mejor adaptadas serán más abundantes, porque sus portadores se reproducirán más eficientemente, las transmitirán a su descendencia y aumentarán su proporción en la población.

A continuación se definen algunos conceptos básicos de la computación evolutiva.

1.1.1. Representación

Se denomina *cromosoma* a una estructura de datos que contiene una cadena de variables de decisión de algún problema (figura 1.1). Usualmente es una cadena binaria, pero también es posible usar una cadena con valores enteros o reales.

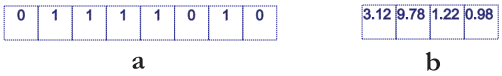

Figura 1.1: Ejemplo de un cromosoma binario (a) y otro real (b).

Llamamos *gene* a una subcadena del cromosoma que codifica, comúnmente, a un solo parámetro de diseño. Estos genes toman ciertos valores, llamados *alelos*, de algún alfabeto genético. De esta manera, si usamos una representacion binaria, los alelos pueden tomar el valor de 0 o de 1. Un *locus* define la posición de un gene dentro del cromosoma (figura 1.2).

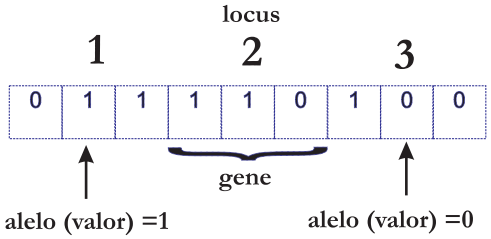

Figura 1.2: Cromosoma binario constituido por tres genes.

Se denomina *genotipo* a la codificación (por ejemplo, binaria, entera o real) de los parámetros de decisión. Mientras que *fenotipo* es la decodificación del genotipo, con el fin de obtener los valores de los parámetros usados como entrada en la función objetivo del problema que se trata (figura 1.3).

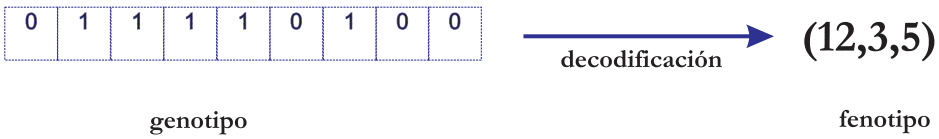

Figura 1.3: Decodificación del genotipo al fenotipo.

Un *individuo* es una solución potencial al problema que se trata. Cada individuo contiene un cromosoma. A un conjunto de individuos se le nombra *población*. La *aptitud* de un individuo es la evaluación de la función de aptitud e indica qué tan bueno es el individuo (es decir, la solución al problema) con respecto a los demás.

1.1.2. Operadores evolutivos

Existe una amplia variedad de operadores que se han empleado en los algoritmos evolutivos. Algunos de ellos están diseñados especialmente para una clase particular de problemas. A continuación se describen los más usados comúnmente:

- **Selección**: la selección determina la probabilidad de elegir un individuo para que produzca descendencia por medio de la recombinación y la mutación. El esquema de selección es una de las partes cruciales de un algoritmo evolutivo, puesto que sesga la búsqueda de manera que eventualmente se llegue a la solución óptima (o su proximidad).

- **Recombinación**: el operador de recombinación (cruza) tiene la finalidad de heredar la información (genes) de dos o más padres a la descendencia. En la computación evolutiva la cruza entre cromosomas se simula intercambiando segmentos de cadenas lineales de longitud fija. Lo usual es que las técnicas de cruza se apliquen sobre representaciones binarias; sin embargo, con las modificaciones adecuadas, se pueden generalizar a alfabetos de cardinalidad mayor.

- **Mutación**: la mutación consiste en pequeñas modificaciones al cromosoma de un individuo. En los algoritmos genéticos la mutación es considerada un operador secundario.

1.1.3. Principales paradigmas

Actualmente existen, principalmente, tres paradigmas inspirados en los principios del *neodarwinismo*: los *algoritmos genéticos*, la *programación evolutiva* y las *estrategias evolutivas*. De manera genérica, a los algoritmos de la computación evolutiva se les llama *algoritmos evolutivos*. Estas tres técnicas tienen en común la reproducción, la variación aleatoria, la competencia y la selección de individuos contendientes dentro de una población.

- **Estrategias Evolutivas**: este modelo enfatiza los nexos conductuales entre padres e hijos, en lugar del nexo genético. Éstas fueron concebidas para construir sistemas capaces de resolver problemas de optimización complejos en los que las variables son números reales. Por ello, la representación natural fue un vector de genes con valores reales, el cual era manipulado, principalmente, por operadores de mutación que perturbaban dichos valores reales [Bäck et al., 1997].

  La primera versión de una estrategia evolutiva, nombrada $(1 + 1)$-EE, crea un solo hijo a partir de un solo padre, y ambos competían para sobrevivir; el peor individuo era eliminado, mientras que el mejor se mantenía para la siguiente generación. En la $(1 + 1)$-EE, a partir de un padre $x^t = (x_1, \ldots, x_n)$, el hijo se genera mediante la expresión:

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

  donde t se refiere a la generación actual, $i = 1, \ldots, n$ es la i-ésima componente de los vectores x, N, σ y σ_i^t es un número aleatorio gaussiano con media cero y desviación estándar σ_i . Los números aleatorios son generados de manera independiente.

- **Programación Evolutiva**: propuesta por Lawrence Fogel consideraba que el comportamiento inteligente requiere de dos habilidades: la primera, a través de un organismo para poder hacer predicciones correctas dentro de su ambiente; y la segunda, es la capacidad de traducir estas predicciones en una respuesta adecuada para una meta dada. Los autómatas de estados finitos fueron la representación ideal para modelar este comportamiento [Fogel, 1964].

- **Algoritmos Genéticos**. Propuestos por [Holland, 1992]. Hay tres características principales que los distinguen de los demás algoritmos evolutivos: la representación de los individuos (cadena binaria); el método de selección (selección proporcional); y el uso de la cruza como el operador principal para modificar al individuo. En este caso, la mutación es un operador secundario.

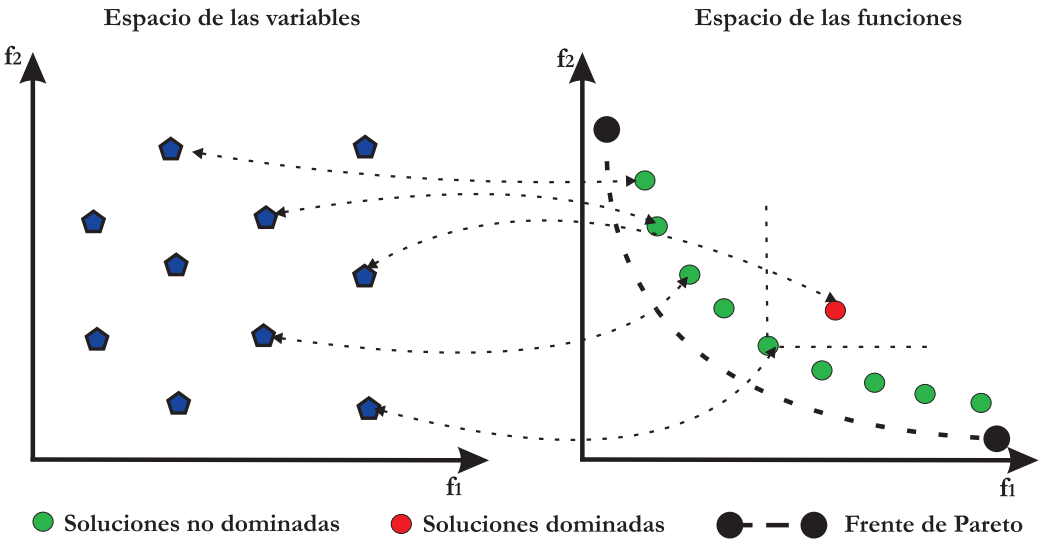

Figura 1.4: Un problema de dos objetivos con dos variables

1.2. Optimización multi-objetivo

Se define el problema de optimización multi-objetivo como: "La tarea de encontrar un vector de variables de decisión que satisfaga restricciones y optimice un vector de funciones objetivo. Esas funciones, generalmente están en conflicto unas con otras y, describen matemáticamente un criterio de desempeño. Por lo tanto, el término optimizar significa encontrar un vector solución con valores aceptables para todas las funciones objetivo" [Osyczka, 1985]. Los problemas multi-objetivo son aquellos en los que se deben optimizar $k \geq 2$ funciones objetivo simultáneamente. El proceso de optimización puede significar la maximización de las k funciones, la minimización de las k funciones o una combinación de maximización y minimización de estas funciones.

Definición 1.1 (Problema de Optimización)**.** Un problema de optimización multi-objetivo se define como la tarea de minimizar (o maximizar)

$$F(\vec{x}) = (f_1(\vec{x}), \ldots, f_k(\vec{x})),$$

donde $k \geq 2$ es el número de funciones a optimizar, sujetas a $g_i(\vec{x}) \leq 0$ (restricciones de desigualdad) y $h_j(\vec{x}) = 0$ (restricciones de igualdad) con $i = \{1, \ldots, m\}$, $j = \{1, \ldots, n\}$ y $\vec{x} \in \Psi$, donde $\vec{x}$ es un vector n-dimensional de variables de decisión $(\vec{x} = (x_1, \ldots, x_n) \in \Psi)$. Las restricciones $g_i(\vec{x}) \leq 0$ y $h_j(\vec{x}) = 0$ deben ser satisfechas al mismo tiempo que se minimiza (o maximiza) $F(\vec{x})$ y Ψ contiene todos los posibles $\vec{x}$ que pueden ser usados para satisfacer la evaluación de $F(\vec{x})$.

El vector de variables de decisión puede ser continuo o discreto mientras que las k funciones pueden ser lineales o no, así como continuas o discretas. La función de evaluación $F : \Psi \rightarrow \Omega$ es una transformación del vector de variables de decisión $\vec{x} = (x_1, \ldots, x_n)$ en un vector de respuesta $\vec{y} = (y_1, \ldots, y_k)$. La figura 1.4 muestra un problema de dos objetivos con dos variables.

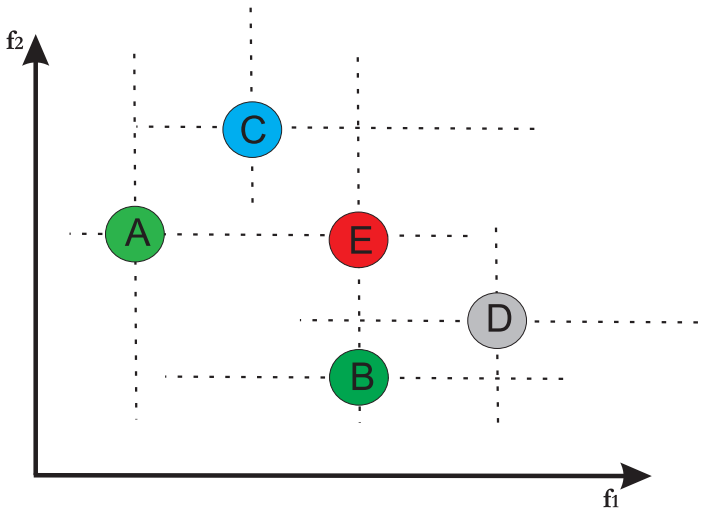

Figura 1.5: Dominancia de Pareto

Cuando el problema es multi-objetivo existe un conflicto que ocasiona que la mejora en algún objetivo provoque el deterioro de otros. Por lo tanto, un problema de optimización multi-objetivo consiste en encontrar el mejor compromiso (balance) entre todos los objetivos. Las soluciones que representan los mejores compromisos entre los objetivos se denominan óptimos de Pareto.

Definición 1.2 (Optimalidad de Pareto)**.** Una solución $\vec{x}^* \in \Psi$ es un óptimo de Pareto con respecto a Ψ si y sólo si no existe $\vec{x} \in \Psi$, para la cual $\vec{v} = F(\vec{x}) = (f_1(\vec{x}), \ldots, f_k(\vec{x}))$ domina a $\vec{u}^* = F(\vec{x}^*) = (f_1(\vec{x}^*), \ldots, f_k(\vec{x}^*))$.

Definición 1.3 (Dominancia de Pareto)**.** Un vector $\vec{u} = (u_1, \ldots, u_k)$ se dice que domina a $\vec{v} = (v_1, \ldots, v_k)$, denotado por $\vec{u} \preceq \vec{v}$, si y sólo si $\vec{u}$ es parcialmente menor que $\vec{v}$, es decir, $\forall i \in \{1, \ldots, k\} : u_i \leq v_i \wedge \exists i \in \{1, \ldots, k\} : u_i < v_i$.

La figura 1.5 muestra gráficamente la dominancia de Pareto para un problema de optimización con dos objetivos en el cual $A \preceq C$ tal que A es mejor en ambas funciones, $A \preceq E$ tal que A es igual con respecto a f_2 pero es mejor con respecto a f_1, $B \preceq E$ tal que B es igual con respecto a f_1 pero mejor con respecto a f_2, $B \preceq D$ tal que B es mejor en ambas funciones, y A y B son incomparables (A y B son óptimos de Pareto).

Definición 1.4 (Conjunto de óptimos de Pareto)**.** Para un problema multi-objetivo determinado $F(\vec{x})$, el conjunto de óptimos de Pareto, denotado por P^* o P_{real}, está definido como:

$$P^* = \{\vec{x} \in \Psi | \quad \nexists \vec{y} \in \Psi \qquad F(\vec{y}) \preceq F(\vec{x})\}$$

Los elementos del conjunto de óptimos de Pareto tienen vectores objetivo que no pueden ser mejorados sin empeorar al menos otro objetivo. Los vectores objetivo de las soluciones óptimas de Pareto se denominan vectores (o soluciones) *no dominados*. Los vectores de las funciones objetivo correspondientes al conjunto de óptimos de Pareto conforman el denominado frente de Pareto ($\mathcal{F}^*$).

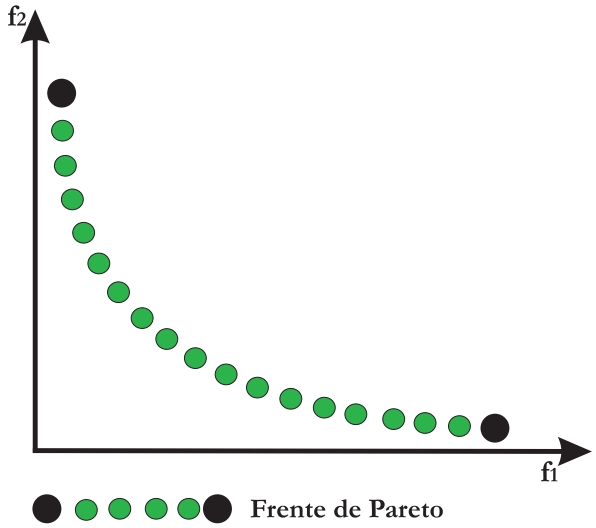

Figura 1.6: Frente de Pareto

Definición 1.5 (Frente de Pareto)**.** Para un problema multi-objetivo determinado $F(\vec{x})$, siendo su conjunto de óptimos de Pareto (P^*) , el Frente de Pareto $\mathcal{F}^*$ o $\mathcal{F}_{real}$, está definido como (ver figura 1.6):

$$\mathcal{F}^* = \{\vec{u} = F(\vec{x}) \mid \ \vec{x} \in P^*\}$$

Definición 1.6 (Vector Ideal y vector de Nadir)**.** Para un problema multi-objetivo determinado $F(\vec{x})$, siendo su conjunto de óptimos de Pareto, P^*, el vector ideal se define como (ver figura 1.7):

$$f_{ideal} = \left(\min_{\vec{x}\in P^*} f_1(\vec{x}), \ldots, \min_{\vec{x}\in P^*} f_k(\vec{x})\right)$$

Si el vector ideal de un problema es alcanzabe, entonces los objetivos del problema no están en conflicto y la solución del problema es única. Por lo tanto, cada objetivo del problema puede ser optimizado por separado, sin requerirse el uso de un algoritmo evolutivo multi-objetivo. De manera similar, el vector de Nadir se define como (ver figura 1.7):

$$f_{nadir} = \left(\max_{\vec{x}\in P^*} f_1(\vec{x}), \ldots, \max_{\vec{x}\in P^*} f_k(\vec{x})\right)$$

Este vector contiene, entonces, los peores valores de las funciones objetivo y suele usarse (junto al vector ideal) para acotar el frente de Pareto.

1.2.1. Métricas para evaluar la eficacia

Al trabajar en optimización multi-objetivo, se tienen que tomar en cuenta dos nociones importantes: la convergencia y la dispersión. Para medir la convergencia, se debe de estimar qué tan lejos están las soluciones que generamos del verdadero frente de Pareto y para medir la dispersión, se debe estimar qué tan uniformemente están distribuidos las soluciones a lo largo del frente de Pareto. Para medir la convergencia se pueden usar medidas como las siguientes:

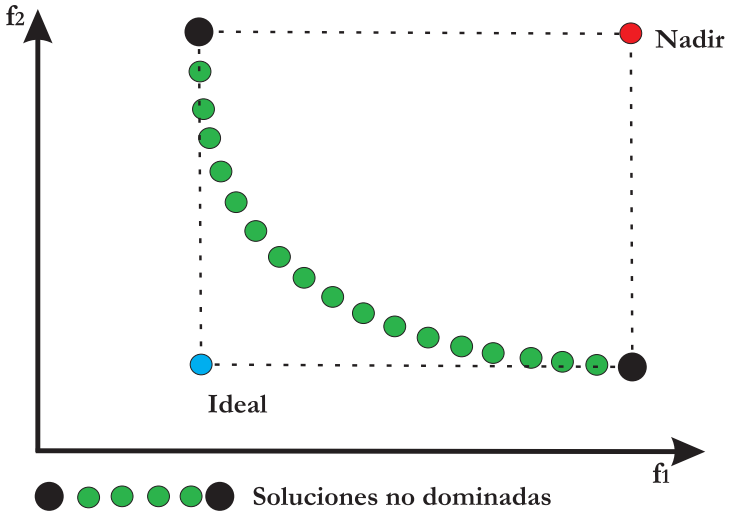

Figura 1.7: Vector Ideal y vector de Nadir

Definición 1.7 (Distancia Generacional Invertida (IGD))**.** Determina cuán lejos, en promedio, se encuentra el verdadero frente de Pareto, del frente obtenido por el algoritmo. Esta métrica evita algunos problemas que posee la denominada Distancia Generacional, en particular, en los casos en los que el frente obtenido posee pocos puntos, pero agrupados en una sola región [Veldhuizen and Lamont, 1998]. Su definición es la siguiente:

$$IGD = \frac{1}{N} \cdot \sqrt{\sum_{i=1}^{N} d_i^2},$$

donde N es la cantidad de puntos del verdadero frente de Pareto, d_i la distancia euclidiana (medida en el espacio de las funciones objetivo) entre cada punto del verdadero frente, al punto más cercano del frente obtenido. Un valor $IGD = 0$ indica que el frente obtenido es el verdadero frente de Pareto, cualquier otro valor indica que el frente obtenido se desvía del verdadero frente de Pareto.

Definición 1.8 (Cobertura de Conjuntos ($\mathcal{C}$))**.** Determina la cobertura relativa de un conjunto, con respecto a otro. Fue propuesta por [Zitzler et al., 2000] para valorar cuantitativamente cuánto un conjunto cubre o domina a otro. Si A y B son dos conjuntos no dominados, la métrica de cobertura de $\mathcal{C}(A, B)$ calcula la fracción de vectores en B que son dominados débilmente por los vectores de A. Se define como:

$$\mathcal{C}\left(A, B\right) = \frac{|\vec{b} \in B; \exists \vec{a} \in A : \vec{a} \preceq \vec{b}|}{|B|}$$

Si todos los vectores en B son dominados o son iguales a los vectores en A, entonces $\mathcal{C}$ es igual a uno. Si ninguno de los vectores en B es dominado o igual a algún vector de A, entonces $\mathcal{C}$ es igual a cero. Lo ideal es que $\mathcal{C}\left(A, B\right)$ y $\mathcal{C}\left(B, A\right)$ se consideren de manera separada, ya que $\mathcal{C}\left(A, B\right)$ no es equivalente a $1 - \mathcal{C}\left(B, A\right)$

Definición 1.9 (Dispersión o Espaciamiento (Spread) (S))**.** Determina la dispersión de los puntos sobre el frente. Fue utilizada por [Deb et al., 2000] para conocer cómo están

distribuidas las soluciones a lo largo del frente de Pareto. Se define como:

$$S = \frac{d_f + d_l + \sum_{i=1}^{N-1} |d_i - d^{'}|}{d_f + d_l + (N-1) \cdot d^{'}}$$

donde N es la cantidad de puntos del frente, d_i es la distancia euclidiana entre soluciones consecutivas, $d^{'}$ es el valor medio de todas las distancias y, d_f y d_l son las distancias euclidianas a los extremos del frente de Pareto. Un valor $S = 0$ indica la distribución ideal (dispersión perfecta).

La dispersión indica qué tan bien están distribuidas las soluciones en el verdadero frente de Pareto (o su aproximación). Este indicador es importante, ya que ofrece más opciones para la toma de decisiones al momento de elegir una o más soluciones no dominadas.

1.2.2. Algoritmos evolutivos multi-objetivo

Hoy en día existen diversas técnicas de programación matemática para resolver problemas de optimización multi-objetivo [Miettinen, 1999, Osyczka, 1985]. Sin embargo, la complejidad de muchos problemas de optimización multi-objetivo del mundo real vuelven a estas técnicas inadecuadas o incluso inaplicables para resolverlos. La complejidad de estos problemas se debe, por ejemplo a: la multimodalidad, la alta dimensionalidad del espacio de búsqueda, a la discontinuidad de las funciones objetivo, a desconexiones tanto en el espacio de la variables de decisión como en el de las funciones objetivo, entre otras causas.

Rosenberg planteó utilizar un método genético de búsqueda para resolver problemas de optimización multi-objetivo por primera vez, pero no llegó a desarrollar un algoritmo, dado que transformó el problema bi-objetivo de su interés en uno mono-objetivo [Rosenberg, 1967]. La primera implementación de lo que actualmente se conoce como un algoritmo evolutivo multi-objetivo fue propueta por [Schaffer, 1985].

Los algoritmos evolutivos resultan particularmente adecuados para resolver problemas de optimización multi-objetivo gracias a que trabajan simultáneamente con un conjunto de soluciones potenciales (es decir, la población). Esta característica les permite encontrar varias soluciones del conjunto de óptimos de Pareto en una sola ejecución. También, son menos sensibles a la forma o continuidad del frente de Pareto. Adicionalmente, son fáciles de usar y no requieren información específica del problema a resolverse.

Los algoritmos evolutivos y los algoritmos evolutivos multi-objetivo son estructuralmente similares. La diferencia principal es que los segundos utilizan un esquema de selección que debe considerar k ($k \geq 2$) funciones objetivo, además de contar con un estimador densidad en el espacio de las funciones objetivo que sesga el mecasnimo de selección de manera que se mantenga diversidad en la población. Sin embargo, el operador de selección espera un solo valor de aptitud. La forma más sencilla de mantener la estructura de un algotirmo evolutivo simple al abordar problemas multi-objetivo (si bien, no es la más adecuada) consiste en transformar el vector de aptitudes en un valor escalar. En el algoritmo 1.1 se describe una estructura básica de un algoritmo evolutivo multi-objetivo, de este tipo. Nótese que este algoritmo no incluye un estimador de densidad.

En general, existen diferentes esquemas para manejar varias funciones objetivo en la selección: los basados en agregación, los basados en poblaciones, y los basados en el concepto de dominancia de Pareto [Zitzler and Thiele, 1999]. Otro componente fundamental de los algoritmos evolutivos multi-objetivo modernos es el elitismo.

Algoritmo 1.1 Estructura básica de un algoritmo evolutivo multi-objetivo agregativo

1: $t \leftarrow 0$.
2: Generar aleatoriamente la población inicial P^t.
3: **Mientras** $t < max_{Gen}$ **Hacer**
4: Para cada individuo $\vec{x} \in P^t$ calcular el valor escalar de aptitud $F(\vec{x})$.
5: Seleccionar de P^t un grupo de padres P'^t basándose en la aptitud.
6: Recombinar los individuos de P'^t para obtener P''^t.
7: Mutar los individuos de $P''(t)$ para obtener P'''^t.
8: $P^{t+1} \leftarrow P'''^t$.
9: $t \leftarrow t + 1$.

Actualmente hay dos formas principales de llevar a cabo la implementación del elitismo en un algoritmo evolutivo multi-objetivo. La primera, es combinar la población anterior y la descendencia o la población nueva, y posteriormente aplicar una selección determinista en lugar de reemplazar la vieja población por la descendencia [Deb et al., 2000]. La segunda, es mantener una población secundaria llamada archivo histórico (población secundaria, población elitista o archivo externo) en el cual se retienen las soluciones no dominadas encontradas en el transcurso del proceso de búsqueda [Zitzler and Thiele, 1999](ver figura 1.8). En muchos problemas de optimización multi-objetivo, el conjunto de óptimos de Pareto es muy grande. Por ello, se debe aplicar una estrategia para decidir la entrada de la nuevas soluciones al archivo y evitar que el archivo crezca indefinidamente.

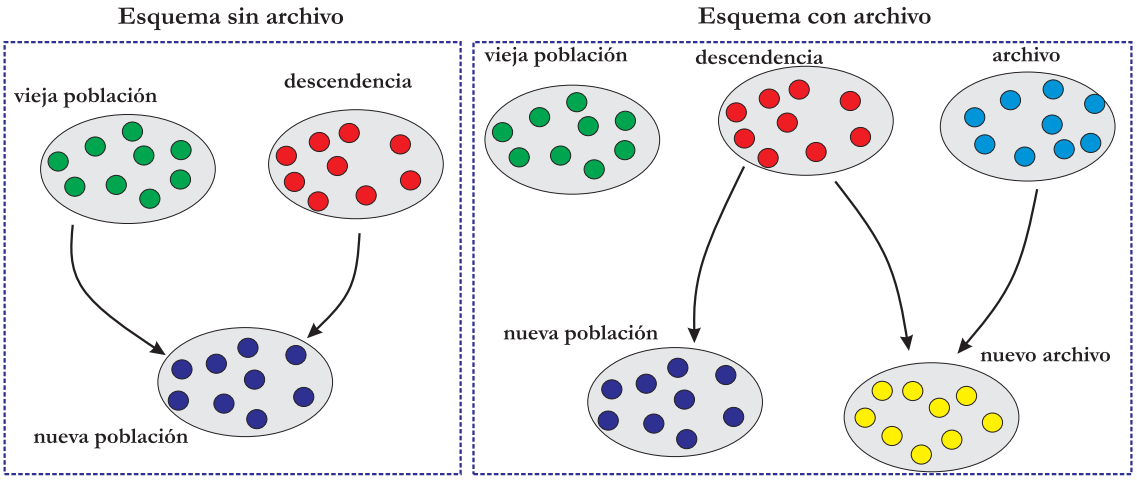

Figura 1.8: Dos esquemas para llevar a cabo el elitismo.

1.2.3. Clasificación

Existen diferentes maneras de clasificar a los algoritmos evolutivos multi-objetivo. Quizás la taxonomía más simple, es la basada en el tipo de mecanismo de selección que usan [Coello et al., 2006]:

- **Enfoques basados en agregación**. Esta técnica simplemente combina todos los objetivos en un valor escalar, es decir, transforman un problema de optimización

multi-objetivo en un problema de optimización mono-objetivo. Para ello se usa la siguiente expresión:

$$\text{mín} \sum_{i=1}^{k} w_i \cdot f_i(\vec{x})$$

donde $w_i > 0$ son los coeficientes de ponderación que representan la importacia de cada cada objetivo k del problema. Usualmente se presupone que:

$$\sum_{i=1}^{k} w_i = 1$$

Las funciones agregativas lineales tienen diversos inconvenientes, entre los que destaca el hecho de no poder generar partes no convexas del frente de Pareto. Las funciones agregativas no lineales no tienen esta limitante, pero no son muy comunes en la literatura especializada. Pese a sus limitantes, las funciones agregativas han sido utilizadas con éxito en problemas combinatorios multi-objetivo.

- **Enfoques basados en población**. En este tipo de enfoque, la población se utiliza para la diversificación de la búsqueda, pero el concepto de dominancia de Pareto no se incorporan directamente en el proceso de selección. Un ejemplo clásico es el algoritmo **VEGA** (*Vector Evaluated Genetic Algorithm*) [Schaffer, 1985]. VEGA consiste en un algoritmo genético con un mecanismo de selección modificado. En cada generación, se crea un número de sub-poblaciones de acuerdo con una selección proporcional a cada objetivo por cada función a la vez. Estas sub-poblaciones son luego mezcladas entre sí para obtener una nueva población, en la que el algoritmo genético aplica los operadores de cruza y mutación. VEGA tiene varios problemas, del cual se puede destacar el que su esquema de selección se opone al concepto de dominancia de Pareto.

- **Enfoques basados en optimilidad de Pareto**. En este tipo de enfoque, se consideran los algoritmos evolutivos multi-objetivo que incorporan el concepto de Pareto en su mecanismo de selección. Los siguentes algoritmos evolutivos son un conjunto representativo que utilizan este enfoque en los últimos años:

  - **PAES (*The Pareto Archived Evolution Strategy*)**: este método consiste en una estrategia evolutiva (1+1), en combinación con un archivo histórico que almacena algunas de las soluciones no dominadas encontradas previamente. Este archivo es usado como un conjunto de referencia contra el cual se compara cada individuo mutado. PAES también utiliza un nuevo enfoque de mantener diversidad, que consiste de un procedimiento de agrupamiento que divide el espacio de las funciones objetivo de una manera recursiva. Cada solución se situa en una rejilla de localización, con base en los valores de las funciones objetivo. Se mantiene un mapa en dicha rejilla y un conteo del número de soluciones que residen en cada retícula de la misma. Dado que el procedimiento es adaptivo, no se requieren parámetros extras en el algoritmo [Angeline et al., 1999].

- **NSGA (*The Nondominated Sorting Genetic Algorithm*)**: este método clasifica a los individuos de la población en varias capas. Antes de hacer la selección, la población se clasifica conforme a su no dominancia: todos los individuos no dominados se clasifican en una categoría con un valor ficticio de aptitud el cual es proporcional al tamaño de la población, para ofrecer el potencial reproductivo de estos individuos [Srinivas and Deb, 1994]. NSGA es un algoritmo no elitista. La siguiente versión de este algoritmo, es el **NSGA-II**, que utiliza el elitismo y un operador de comparación que clasifica a la población conforme a la dominancia de Pareto y la densidad de la región. Por lo que, este algoritmo es más eficiente y efectivo que su predecesor.
- **SPEA (*The Strength Pareto Evolutionary Algorithm*)**: este método introduce elitismo al almacenar a los individuos no dominados en un archivo. Para cada individuo de este conjunto externo se calcula un valor llamado fortaleza (*strength*). Este valor es proporcional al número de soluciones que domina un cierto individuo del conjunto externo. La aptitud de los miembros de la población actual se calcula de acuerdo a la fortaleza de los individuos de la población externa que lo dominan. En este esquema el objetivo es minimizar la aptitud, es decir, los valores pequeños de aptitud corresponden a altas probabilidades de reproducción. El esquema para asignar la aptitud pretende conseguir que la búsqueda se dirija hacia el frente de Pareto y a la vez preservar la diversidad de las soluciones no dominadas. Para proveer diversidad a la población, se utiliza una técnica de cúmulos (*clustering*), llamada método de enlace promedio [Zitzler and Thiele, 1999].

  La siguiente versión de este algoritmo, es el **SPEA2**, el cual tiene las siguientes diferencias [Zitzler et al., 2002]:

  - Incorpora una estrategia de asignación de aptitud de grano-fino, que por cada individuo toma en cuenta el número de individuos a los que domina y el número de individuos por los cuales es dominado.
  - Utiliza una técnica de estimación del vecino más cercano que guía la búsqueda de una manera más efectiva.
  - Presenta un método mejorado de truncamiento del archivo, que garantiza la preservación de las soluciones de los extremos del frente de Pareto.
  - Utiliza un algotitmo de *clustering* de menor complejidad.

Como se mostró anteriormente en la literatura existen muchas formas de poder resolver problemas multi-objetivo utilizando diversos enfoques. En este trabajo de investigación se plantea comparar el desempeño del algoritmo propuesto con respecto a las soluciones obtenidas con los siguientes algoritmos reprensetativos de la literatura especializa:

- El primero, es el que presentan [Raquel and Jr., 2005] en el cual se plantea el uso de un algoritmo de cúmulos de partículas para problemas multi-objetivo (*An effective use of Crowding Distance in Multiobjetive Particle Swarm Optimization*, **MOPSOcd**) mediante la incorporación del mecanismo del cálculo de la distancia de *Crowding* específicamente para seleccionar al mejor líder global y en el método para mantener las soluciones no dominadas en el archivo externo. El algoritmo 1.2 muestra la estructura básica del MOPSOcd.

Las características que enmarcan al algoritmo MOPSOcd son:

- Hace uso de un operador de mutación junto a la distacia de *Crowding* para mantener la diversidad de soluciones no dominadas en un archivo externo.
- La selección del mejor líder global se hace de entre aquellas soluciones no dominadas con los valores más altos de la distancia de *crowding* (por ejemplo, 10 % de la parte superior del archivo ordenado de manera desdente).
- La selección de diferentes guías para cada partícula en una parte determinada del archivo de soluciones no dominadas ordenadas en función de las distancia de *crowding* (por ejemplo, 10 % de la parte inferior del archivo) permite que las partículas de la población primaria avancen hacía las soluciones no dominadas del archivo externo con el menor agrupamiento en el espacio de las funciones objetivo.
- Los resultados muestran que este algoritmo es competitivo en lo que a convergencia hacia el frente de Pareto se refiere, generando también un conjunto bien distribuido de soluciones no dominadas.
- Adapta un mecanismo de manejo de restricciones usado por el NSGA-II, debido a su simplicidad en la factibilidad y la comparación de soluciones no dominadas.

- El segundo, es un algoritmo cuyo mecanismo de selección adopta el hipervolumen, llamado *S metric selection evolutionary multiobjective algorithm* (**SMS-EMOA**) propuesto por [Beume et al., 2007]. Este algoritmo genera en cada iteración nuevas soluciones por medio de operadores de variación aleatoria. Por tanto, las soluciones se descartan conforme a su contribución al hipervolumen. Los resultados muestran que el SMS-EMOA es muy potente para problemas difíciles. Su principal inconveniente es su alta complejidad computacional, que está relacionada con el cálculo de la contribución al hipervolumen que puede tomar mucho tiempo conforme se aumenta el número de objetivos. El algoritmo 1.3 muestra la estructura básica del SMS-EMOA.

  Los pilares principales que enmarcan al algoritmo SMS-EMOA son:

  - El ordenamiento no dominado con base en los niveles o jerarquías de no dominancia (*rank*) usado por el NSGA-II.
  - El hipervolumen se aplica como criterio de selección para descartar individuos.

- Por último, pero no menos importante, está el **NSGA-II** el cual ha sido muy popular en la literatura especialzada debido, sobre todo, a su eficiencia y facilidad de uso. Este algoritmo tiene también un bajo costo computacional, basándose en un operador de agrupamiento (*Crowding*) y un esquema de selección que compara la población de padres con la de hijos quedándose con los mejores individuos de entre todos ellos [Deb et al., 2000]. El algoritmo 1.4 muestra la estructura básica del NSGA-II.

  Las características que enmarcan al algoritmo NSGA-II son:

  - El ordenamiento no dominado mediante una técnica de comparación que utiliza una subpoblación auxilizar.

Algoritmo 1.2 Estructura básica del MOPSOcd

Entrada: Problema de Optimización F.

Salida: Soluciones no dominadas en un archivo acotado A.

1: Inicializar la velocidad $\vec{v} \leftarrow 0$ del cúmulo S.
2: Inicializar aleatoriamente las posiciones del cúmulo S.
3: Evaluar las partículas del cúmulo S de acuerdo a las funciones objetivo.
4: Seleccionar el $pBest$ inicial de las partículas.
5: Seleccionar el $g\vec{B}est$ como el mejor encontrado en el cúmulo S.
6: Inicializar el contador de generaciones con $t \leftarrow 0$.
7: Insertar las partículas no dominadas de la población en el archivo A.
8: **Mientras** $t < max_{Gen}$ **Hacer**
9: Calcular la distancia de *crowding* de cada partícula no dominada en el archivo A
10: Ordenar las partículas no dominadas en el archivo A de manera descendiente de acuerdo a los valores de *crowding*.
11: **Para** $i \leftarrow 1$ hasta tam_S **Hacer**
12: Seleccionar aleatoriamente el $g\vec{B}est$ desde una porción especificada (por ejemplo, 10 % de la parte superior) del archivo ordenado A.
13: Generar una nueva velocidad
$\vec{v}_i^{t+1} \leftarrow \omega \cdot \vec{v}_i^t + \phi_1 \cdot rnd_1 \cdot \left(p\vec{B}estt_i^t - \vec{S}_i^t\right) + \phi_2 \cdot rnd_2 \cdot \left(g\vec{B}est - \vec{S}_i^t\right)$
14: Calcular las nuevas posiciones
$\vec{S}_i^{t+1} \leftarrow \vec{S}_i^t + \vec{v}_i^{t+1}$
15: Reintegrar la partícula i si no pertenece a la región de búsqueda y la velocidad de esta partícula se multiplica por -1.
16: **Si** $t < max_{Gen} \cdot pMut$ **Entonces**
17: Aplicar el operador de mutación en la partícula i.
18: Evaluar la partícula i de acuerdo a las funciones objetivo.
19: Insertar todas las pastículas no dominadas de S en A si no son dominadas por alguna partícula ya almacenada. Todas las partículas en el archivo que sean dominadas por la nueva partícula son eliminadas. Si el archivo esta lleno, las partículas se reemplazan de acuerdo al siguiente criterio:

- Calcular la distancia de *Crowding* de cada partícula no dominada del archivo A.
- Ordenar las partículas no dominadas en el archivo A de manera descendiente de acuerdo a los valores
- Seleccionar aleatoriamente una partícula desde una porción espeficada (por ejemplo, 10 % de la parte inferior) del archivo ordenado A, las cuales comprenden las partículas más agrupadas en el archivo y luego sustituirla por la nueva partícula.

20: Actualizar el $pBest$ de cada partícula en S. Si el actual $pBest$ domina la posición en memoria, la posición de la es actualizada utilizando $pBest \Leftarrow S$
21: $t \leftarrow t + 1$.

Algoritmo 1.3 Estructura básica del SMS-EMOA

Entrada: Problema de Optimización F.
Salida: Una aproximación al frente de Pareto.

1: $t \leftarrow 0$
2: Inicializar la población aleatoriamente P_t
3: **Mientras** criterio de paro **Hacer**
4: Generar un nuevo individuo x de acuerdo con un operador de variación aleatoria.
5: $Q_{t+1} \leftarrow P_t \cup \{x\}$.
6: Calcular el frente de Pareto de acuerdo con la asignación de valores de aptitud con base en los niveles o jerarquías de no dominancia ($rank$) de Q_{t+1}.
7: $W \leftarrow$ individuos con la peor clasificación del frente.
8: $r \leftarrow$ individuos (en W) que menos contribuyen al valor del hipervolumen cubierto por W.
9: $P_{t+1} \leftarrow Q_{t+1} \diagup r$, eliminar los peores inviduos detectados de la población.
10: $t \leftarrow t + 1$.

- El uso de una técnica de agrupación, que no requiere especificar parámetros adicionales para la preservación de la diversidad en la población.
- La asignación de valores de aptitud con base en los niveles o jerarquías de no dominancia ($rank$), aunque se considera en el procedimiento de asignación de valores de distancia de *crowding* utilizados para evaluar la diversidad de las soluciones.

Algoritmo 1.4 Estructura básica del NSGA-II

Entrada: Problema de Optimización F.
Salida: Una aproximación al frente de Pareto.

1: $t \leftarrow 0$
2: Inicializar la población P_t
3: **Mientras** $t < max_{Gen}$ **Hacer**
4: Aplicar cruza y mutación a P_t, para generar Q_t
5: Evauar población Q_t de acuerdo a las funciones objetivo
6: $R_t \leftarrow P_t \cup Q_t$
7: Asignar gerarquias con base en la dominacia de Pareto a R_t
8: Asignar distancia de agrupamiento a R_t
9: Seleccionar los N individuos de R_t, de acuerdo al operador de comparación de agrupamiento $\prec_n$ para obtener P_{t+1}.
10: $tt + 1$

A continuación se describe más a detalle el algoritmo basado en cúmulos de partículas.

1.3. Algoritmo basado en cúmulos de partículas

El algoritmo de optimización mediante cúmulos de partículas fue propuesto en 1995 usándose originalmente para el balance de pesos de una red neuronal. A continuación se

describen las principales componentes de la heurística, los modelos mono-objetivo más característicos y, finalmente, como llevar ésta misma a su versión multi-objetivo.

1.3.1. Paradigma

Un algoritmo basado en cúmulos de partículas o *Particle Swarm Optimization (PSO)* es una técnica inspirada en el comportamiento social del vuelo de las aves o el movimiento de los bancos de peces que intentan encontrar comida. Se fundamenta en los factores que influyen en la toma de decisiones de una partícula que forma parte de un cúmulo de partículas similares. La decisión de cada partícula se realiza conforme a una componente social y una componente individual, mediante las que se determina el movimiento (dirección) para alcanzar una nueva posición en el espacio de soluciones. La metáfora se puede resumir de la siguiente forma: "los individuos que conviven en una sociedad tienen una opinión que es parte de un conjunto de creencias (el espacio de búsqueda) compartido por todos los posibles individuos" [Eberhart et al., 2001].

Siguiendo ciertas reglas de interacción, las partículas en la población adaptan sus esquemas de creencias al de las partículas con más éxito de su entorno. Con el tiempo, surge una cultura de creencias estrechas entre los individuos. Cada solución (partícula) es un ave en el espacio de búsqueda que está siempre en continuo movimiento y que nunca muere. El cúmulo de partículas (*Swarm*) es un sistema multiagente; es decir, las partículas son agentes simples que se mueven por el espacio de búsqueda y que guardan (y posiblemente comunican) la mejor solución que han encontrado hasta el momento. Cada partícula tiene una posición y un vector velocidad que dirige su movimiento. El movimiento de las partículas por el espacio está guiado por las partículas óptimas en el momento actual.

Esta heurística posee las siguientes caracterítisticas [Kennedy and Eberhart, 1995]: En primer lugar, asume un intercambio de información (interacciones sociales) entre las partículas de búsqueda. Es decir, las partículas modifican su dirección en función de las direcciones de las partículas de su vecindario. Además, almacena información histórica de la experiencia propia de cada partícula (influencia individual). Cada partícula decide su nueva dirección en función de la mejor posición por la que pasó anteriormente. Suele tener una convergencia rápida a buenas soluciones.

También comparte otras características con los algoritmos evolutivos, tales como:

- La población es inicializada aleatoriamente y evoluciona iterativamente buscando la mejor solución posible.
- Realiza una búsqueda ciega, es decir, no dispone de ningún conocimiento específico del problema, de manera que la búsqueda se basa exclusivamente en los valores de la función objetivo.
- Trabaja con información codificada (binaria, entera o real), no directamente sobre el dominio del problema, sino sobre una representación de sus elementos.
- Es una técnica de búsqueda estocástica.

En particular, cada partícula puede modificar su propia dirección de búsqueda basándose en tres factores:

- Su conocimiento sobre el entorno.

- Su conocimiento histórico o experiencias anteriores.
- El conocimiento histórico o experiencias anteriores de las partículas situadas en su vecindario.

Sin embargo, no tiene operadores de variación tales como la mutación (como un algoritmo genético [Yang, 2008]) o la cruza, sino que tiene operadores de movimiento. Además, no se crean nuevas partículas sino que las mismas partículas iniciales son modificadas a lo largo del proceso de búsqueda.

A continuación se describen los principales elementos que caracterizan a una partícula.

1.3.2. Estructura de una partícula

La estructura básica de una partícula consta de los siguientes componentes:

- Un vector $\vec{x}$ que describe la ubicación de la partícula dentro del espacio de soluciones. El tamaño de este vector depende del número de variables necesarias para resolver el problema.
- **Valor objetivo**, representa la calidad de la solución representada por el vector $\vec{x}$, obtenido a través del cálculo de la *función de evaluación* o *función objetivo* correspondiente al problema específico.
- Un vector $\vec{v}$ que representa la *velocidad* de la partícula. Este vector, al igual que $\vec{x}$, se modifica en cada iteración del algoritmo, reflejando así el cambio de dirección que sufre la partícula dentro del espacio de búsqueda.
- ***pBest***, mejor valor objetivo encontrado por la partícula hasta el momento.
- ***gBest***, mejor valor objetivo encontrado por las partículas del cúmulo. Este valor está presente si se utiliza un modelo global, es decir, un modelo en el que los individuos son influenciados por el mejor de toda la población.
- ***lBest***, mejor valor objetivo encontrado en el vecindario de una partícula. Este valor está presente si se utiliza un modelo local, es decir, un modelo en el que los individuos son influenciados por el mejor de un grupo de individuos (vecindario). La determinación del vecindario depende de la forma en la que se efectúa el agrupamiento de individuos de la población.

1.3.3. Trayectoria de una partícula

La trayectoria de la partícula dentro del espacio de búsqueda está definida con base en el vector de ubicación $\vec{x}$ y el de velocidad $\vec{v}$, los cuales son sumados para obtener la nueva dirección. Justamente estos cambios de trayectoria son los que determinan la característica principal de la heurística PSO, ya que a través de los mismos las partículas son forzadas a buscar soluciones en las áreas más prometedoras del espacio de búsqueda.

Más específicamente, en cada generación o iteración, la ubicación de cada individuo o partícula es afectado por el movimiento del mejor de toda la población o del vecindario al que pertenezca. La trayectoria de las partículas está definida por la velocidad asociada a

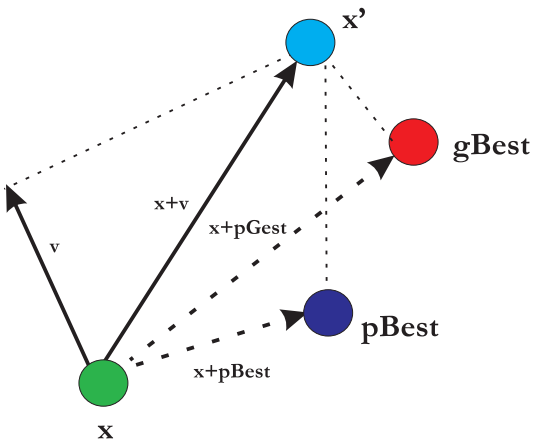

Figura 1.9: Trayectoria de la partícula $\vec{x}$

cada una de ellas, y es la que regula la exploración del espacio de búsqueda. Cada partícula recorre el espacio ayudada por dos valores adicionales: el mejor alcanzado por la partícula hasta el momento (***pBest***) y el mejor alcanzado por todas las partículas de la población (***gBest*** o ***lBest*** si se consideran vecindarios). De esta manera se efectúa una búsqueda "guiada" por los espacios más prometedores. Estos valores aseguran que el tamaño de paso con que se mueven las partículas dentro del espacio sea variable, asegurando que las mismas no "caigan en una rutina", es decir, que no se muevan siempre con el mismo paso. La figura 1.9 ilustra la obtención de la nueva posición $\vec{x}'$ como consecuencia de la suma de la velocidad y los valores adicionales.

La ecuacion (1.1) muestra de manera más formal como una partícula se mueve desde una posición del espacio de búsqueda hasta otra, simplemente, añadiendo al vector posición $\vec{x}_i^t$ el vector velocidad $\vec{v}_i^t$ para obtener un nuevo vector posición, en la generación t:

$$\vec{x}_i^{t+1} \leftarrow \vec{x}_i^t + \vec{v}_i^t \tag{1.1}$$

El vector velocidad de cada partícula es modificado utilizando la velocidad anterior, un componente cognitivo o individual y un componente social. El modelo matemático resultante y que representa el corazón de esta heurística, viene representado por la siguiente ecuación:

$$\vec{v}_i^{t+1} \leftarrow \vec{v}_i^t + \phi_1 \cdot rnd_1 \cdot \left(p\vec{B}est_i^t - \vec{x}_i^t\right) + \phi_2 \cdot rnd_2 \cdot \left(\vec{g}_i^t - \vec{x}_i^t\right) \tag{1.2}$$

La ecuación (1.2) actualiza el vector velocidad de cada partícula i en cada generación t. El componente cognitivo está modelado por el factor $\phi_1 \cdot rnd_1 \cdot \left(p\vec{B}est_i - \vec{x}_i^t\right)$ y representa la distancia entre la posición actual y la mejor conocida por esa partícula, es decir, la decisión que tomará la partícula influenciada por su propia experiencia a lo largo de su vida. El componente social está modelado por $\phi_2 \cdot rnd_2 \cdot \left(\vec{g}_i - \vec{x}_i^t\right)$ y representa la distancia entre la posición actual y la mejor posición del vecindario, es decir, la decisión que tomará la partícula según la influencia que el resto del cúmulo ejerce sobre ella.

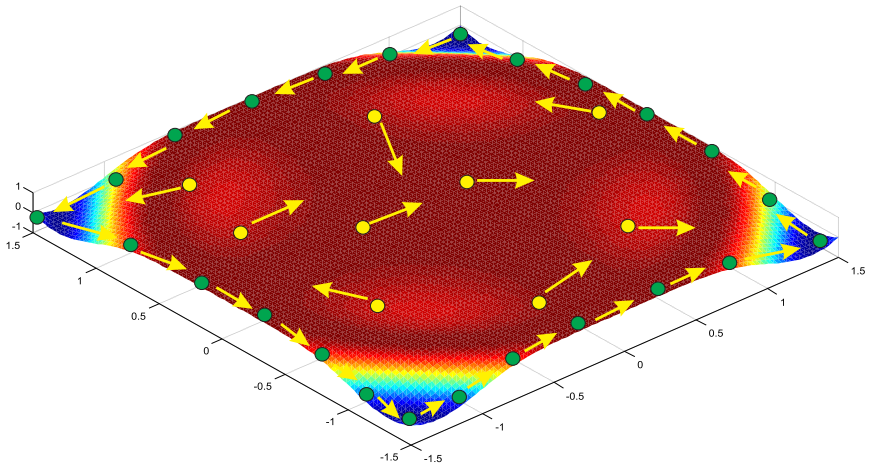

Figura 1.10: Ejemplo de la inicialización del cúmulo. Las partículas se mueven a través de un proceso iterativo.

1.3.4. Descripción de la versión básica

El cúmulo se inicializa generando aleatoriamente las posiciones (ver figura 1.10) y las velocidades iniciales de las partículas. Una vez generadas las posiciones, se calcula el valor de la función objetivo de cada una y se actualizan los valores iniciales de $\vec{x}^t$ y $p\vec{B}est^t$, en la generación $t = 0$. El algoritmo 1.5 muestra una versión básica de PSO [Eberhart et al., 2001], que puede usarse para minimización o maximización. Por conveniencia se representa la mejor partícula del vecindario como $\vec{g}^t$, pudiendo ser ésta $g\vec{B}est^t$ o $l\vec{B}est^t$ si se consideran vecindarios.

Algoritmo 1.5 Pseudocódigo del algoritmo de PSO básico

Entrada: Función a optimizarse $f(\vec{x})$.
Salida: La mejor solución encontrada.

1: Iniciar el contador de generación es $t = 0$.
2: Iniciar aleatoriamente las posiciones $\vec{x}_i$ y $\vec{v}_i$ de las n partículas del cúmulo S.
3: Evaluar la función objetivo con las posiciones $\vec{x}^t$.
4: Encontrar $p\vec{B}est^t$.
5: Encontrar $\vec{g}^t$.
6: **Mientras** $t < maxG$ **Hacer**
7: **Para** $i \leftarrow 1$ hasta $size(S)$ **Hacer**
8: Generar una nueva velocidad.
$\vec{v}_i^{t+1} \leftarrow \vec{v}_i^t + \phi_1 \cdot rnd_1 \cdot \left(p\vec{B}est^t - x_i^t\right) + \phi_2 \cdot rnd_2 \cdot \left(\vec{g}^t - \vec{x}_i^t\right)$
9: Calcular las nuevas posiciones.
$\vec{x}_i^{t+1} \leftarrow \vec{x}_i^t + \vec{v}_i^{t+1}$
10: Evaluar la función objetivo con las nuevas posiciones $\vec{x}^{t+1}$.
11: Encontrar $p\vec{B}est^t$.
12: Encontrar $\vec{g}^t$.
13: $t = t + 1$.
14: **Retornar** La mejor solución encontrada.

En este algoritmo se pueden observar dos partes importantes: la primera es la *inicialización* del cúmulo que asigna a cada partícula de la población un valor aleatorio. La segunda es la *búsqueda* en la cual, el cúmulo recorre el espacio de búsqueda para hallar la mejor solución posible.

Siguiendo este proceso iterativo, se actualizan la velocidad y posición con base en los valores adicionales $pBest$ y g; los valores de las funciones objetivo correspondientes a $pBest$ y g son recalculados y éstos se actualizan sólo si se mejoraron con respecto a los valores previos. Existen dos maneras para actualizar los valores $pBest$ y g: la primera es la forma síncrona, es decir, la actualización de los mejores valores encontrados se realiza en forma separada de la actualización de las posiciones de las partículas. La otra forma es la actualización asíncrona, donde la única diferencia es la ubicación de las actualizaciones de los mejores individuos. El modelo asíncrono mantiene actualizadas instantáneamente a las mejores partículas del cúmulo, mientras que en la versión síncrona se realiza la actualización sólo una vez por iteración.

1.3.5. Topologías del cúmulo de partículas

Un aspecto muy importante a considerar es la manera en la que una partícula interacciona con las demás partículas de su vecindario. El desarrollo de una partícula depende tanto de la topología del cúmulo como de la versión del algoritmo. Las topologías definen el entorno de interacción de una partícula individual con su vecindario. La propia partícula siempre pertenece a su entorno, y éste puede ser de dos tipos (ver figura 1.11) [Eberhart et al., 2001]:

- **Geográfico**: se calcula la distancia de la partícula actual al resto y se toman las más cercanas para componer su entorno.
- **Social**: se define previamente una lista de vecinos para la partícula, independientemente de su posición en el espacio.

Todos los individuos pertenecientes a la misma estructura social comparten información, aprenden y siguen a los mejores dentro de su propia estructura. Por lo tanto, es posible configurar varios tipos de topologías o interacciones dentro de un entorno social del cúmulo (ver figura 1.12):

- **Global ($gBest$ o totalmente conectados)** [Kennedy and Eberhart, 1995]: se establece un vecindario global en el que toda partícula es vecina de la totalidad del cúmulo favoreciendo la explotación del espacio de soluciones.
- **Local ($lBest$)** [Kennedy and Eberhart, 1995]: se establece un vecindario local y cada partícula es conectada con sus vecinas inmediatas en el cúmulo. Esta topología ofrece la ventaja de establecer subcúmulos que realizan búsquedas en diversas regiones del espacio del problema y de esta forma se favorece la exploración.
- **Circular o Anular** [Eberhart et al., 2001]: cada individuo está conectado con sus n vecinos inmediatos, intentando imitar al mejor de ellos. En este tipo de estructura, cada vecindario facilita el intercambio de información y, de esta manera, converge a una única solución.

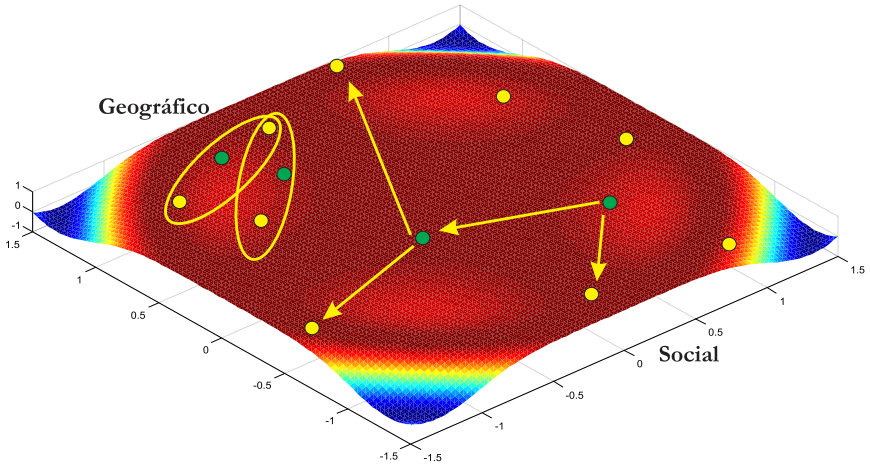

Figura 1.11: Ejemplo de entornos geográficos y sociales. Los entornos sociales son los más empleados. Una vez defínido un entorno, es necesario definir su tamaño; son habituales valores de 3 y 5 para el tamaño del cúmulo pues con esos valores, PSO suele tener un buen comportamiento. Obviamente, cuando el tamaño es todo el cúmulo de partículas, el entorno es a la vez geográfico y social, obteniéndose así un PSO Global.

- **Cuadrada** [Kennedy and Mendes, 2006]: se dispone el cúmulo como una matriz rectangular, donde cada partícula es conectada con las partículas de manera toroidal.

- **Arcos aleatorios** [Cagnina, 2010]: para n partículas se asignan n conexiones simétricas entre pares de individuos. Como su nombre lo indica, las conexiones son asignadas de forma aleatoria entre las n partículas.

- **Pirámide** [Cagnina, 2010] : la idea de esta topología es la creación de una estructura tridimensional que comunique a las partículas. Se asemeja a un modelo de alambre tridimensional.

- **Vacío** [Reyes Sierra and Coello Coello, 2006]: se establece que las partículas están aisladas, es decir, cada partícula está conectada sólo consigo misma.

- **Red estrella** [Reyes Sierra and Coello Coello, 2006]: en esta topología una partícula está conectada a todas las demás. Las partículas están conectadas a una sola central que tiene toda la información.

- **Árbol** [Eberhart et al., 2001]: cada partícula es influenciada por su mejor posición hasta el momento y por la partícula con la que se encuentre conectada por encima del árbol.

El objetivo de utilizar estructuras de vecindarios es evitar la convergencia prematura del algoritmo hacia óptimos locales. Esto es posible porque en una topología de vecindario, cada individuo es influenciado por el mejor valor objetivo encontrado por grupos más pequeños de partículas, y no por el mejor valor hallado por algún individuo de la población entera.

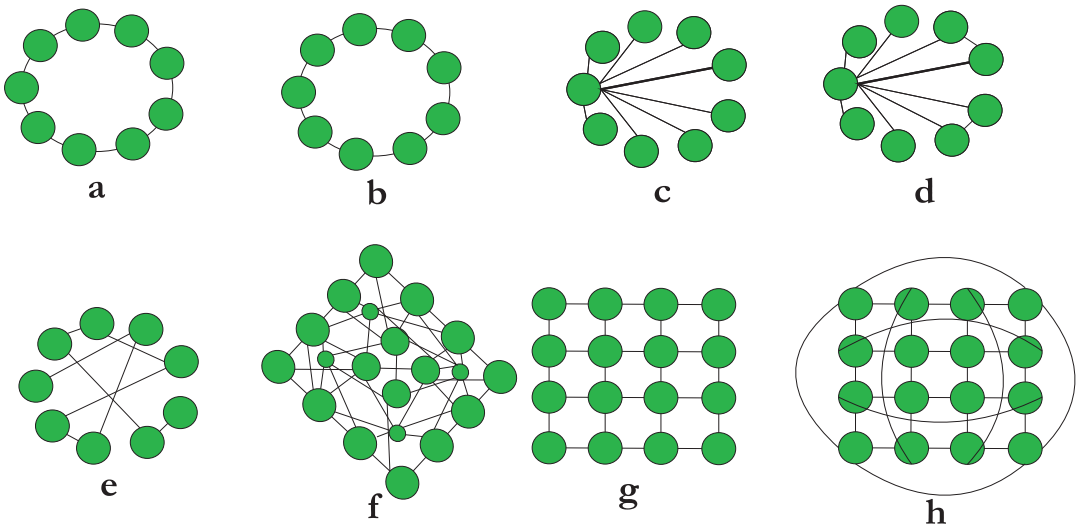

Figura 1.12: (a) Topología circular. (b) Topología circular con conexiones variantes (c) Topología de anillo. (d) Topología de rueda con conexiones variantes. (e) Topología de arcos aleatorios. (f) Topología piramidal. (g) Topología cuadrada. (h) Topología toroidal.

1.3.6. Aspectos avanzados del algoritmo

Se han propuesto varias modificaciones a la version básica de PSO (ver algoritmo 1.5, página 21), de manera que se mejore la velocidad de convergencia y la calidad de las soluciones encontradas. A continuación se describen brevemente algunas variantes del algoritmo original.

Control de velocidad

Un objetivo importante es lograr un balance entre la explotación y exploración del espacio de búsqueda. En esta heurística la tarea recae en la velocidad de las partículas. Si no se controla este elemento, podría suceder que las partículas escapen del espacio de búsqueda del problema, resultando esto en la divergencia de los individuos. Por este motivo es común limitar el crecimiento de la velocidad utilizando algún método [Eberhart et al., 1996]. El más simple es emplear un parámetro de velocidad máxima (elegido dependiendo el problema), de manera que la actualización se efectúa siguiendo la ecuación (1.3) en vez de la (1.2):

$$\vec{v}_i^t = \begin{cases} \vec{v}_i^{t+1} & \text{si} \quad \vec{v}_i^{t+1} < \vec{v}_{\text{máx}} \\ \vec{v}_{\text{máx}} & \text{si} \quad \vec{v}_i^{t+1} \geq \vec{v}_{\text{máx}} \end{cases} \tag{1.3}$$

donde $\vec{v}_i^t$ es la nueva velocidad, definida como: $\vec{v}_i^{t+1}$ (calculada con la ecuación (1.2)) o $\vec{v}_{max}$, dependiendo de la condición. El valor de $\vec{v}_{max}$ nos permite controlar la granularidad de la búsqueda escalando las velocidades. Valores grandes facilitan la exploración del espacio, mientras que los valores más chicos aseguran la explotación, aunque si son demasiado pequeños podría convergerse a óptimos locales.

Factor de Constricción

El factor de Constricción consta de un conjunto de ecuaciones lineales que derivan en la convergencia óptima de las partículas. En este modelo, la velocidad está restringida por una constante χ, con el objeto de asegurar la convergencia a un punto estable del espacio [Clerc, 1999]. La ecuación (1.2) de velocidad se reemplaza por la (1.4).

$$\vec{v}_i^{t+1} = \chi \left(\vec{v}_i^t + \phi_1 \cdot rnd_1 \cdot \left(p\vec{B}est_i^t - \vec{x}_i^t \right) + \phi_2 \cdot rnd_2 \cdot \left(\vec{g}_i^t - \vec{x}_i^t \right) \right) \tag{1.4}$$

donde el factor χ se define como lo muestra la ecuación (1.5).

$$\chi = \frac{2 \cdot K}{|2 - \varphi - \sqrt{\varphi\,(\varphi - 4)}|} \tag{1.5}$$

siendo

$$\begin{aligned} \varphi &= \varphi_1 + \varphi_2 \\ \varphi &\geq 4 \\ \varphi_1 &= \phi_1 \cdot rnd_1 \\ \varphi_2 &= \phi_2 \cdot rnd_2 \end{aligned}$$

con $K \in [0, 1]$. El parámetro K controla la exploración y explotación del cúmulo. Para valores de K cercanos a cero, se obtiene una rápida convergencia con explotación local. Para valores de K cercanos a uno, la convergencia se produce más lentamente y se experimenta un grado mayor de exploración.

Factor de Inercia

El factor de Inercia (ω) fue introducido por [Shi and Eberhart, 1998a] en la fórmula de actualización de la velocidad de las partículas. Inicialmente fue utilizado como mecanismo de control de exploración y explotación del cúmulo. Su objetivo es que la velocidad no exceda los límites establecidos. El factor multiplica a la velocidad previa en la fórmula básica de velocidad y, puede permanecer fijo o ser linealmente decrementado en cada generación. La ecuación (1.2) de velocidad se reemplaza por la (1.6), mientras que la de actualización de posiciones (ecuación 1.1) permanece sin cambios en esta variante.

$$\vec{v}_i^{t+1} = \omega \cdot \vec{v}_i^t + \phi_1 \cdot rnd_1 \cdot \left(p\vec{B}est_i^t - \vec{x}_i^k \right) + \phi_2 \cdot rnd_2 \cdot \left(\vec{g}_i^t - \vec{x}_i^t \right) \tag{1.6}$$

El valor de ω es importante para la convergencia del algoritmo. Para valores $\omega \geq 1$ las velocidades se van incrementando en cada ciclo de vuelo acelerando las partículas al máximo valor permitido, lo que produce la divergencia del cúmulo (incremento de la diversidad). Para valores $\omega < 1$, las partículas se desaceleran hasta velocidad casi cero (explotación). Generalmente el mejor valor para ω es dependiente del problema [Shi and Eberhart, 1998b].

Modelos de configuración

Se pueden obtener diferentes modelos atendiendo a diversos factores de configuración, por ejemplo, según la importancia de los pesos cognitivo y social, y según el tipo de vecindario utilizado. Dependiendo de la influencia de los factores cognitivo y social (valores ϕ_1 y ϕ_2 , respectivamente) sobre la dirección de la velocidad que toma una partícula en el movimiento, se identifican cuatro tipos de modelos [Kennedy, 1997]:

- **Modelo completo**: $\phi_1 > 0$, $\phi_2 > 0$. Tanto el componente cognitivo como el social intervienen en el movimiento.
- **Modelo cognitivo**: $\phi_1 > 0$ y $\phi_2 = 0$. Únicamente el componente cognitivo interviene en el movimiento.
- **Modelo social**: $\phi_1 = 0$ y $\phi_2 > 0$. Únicamente el componente social interviene en el movimiento.
- **Modelo social exclusivo**: $\phi_1 = 0$, $\phi_2 > 0$ y $\vec{g} = \vec{x}_i$. La posición de la partícula en sí no puede ser la mejor de su entorno.

Los factores cognitivo y social no son críticos para la convergencia. Sin embargo, el valor correcto de estos factores da lugar a una convergencia más rápida y a la reducción de los mínimos locales. Existe evidencia que indica elegir un factor más cognitivo (ϕ_1) que un factor social (ϕ_2), con $\phi_1 + \phi_2 < 4$ [Clerc and Kennedy, 2002]. A continuación, se hace una descripción de cómo extender el algoritmo PSO para problemas multi-objetivo.

1.3.7. Descripción de la versión multi-objetivo

Extender el algoritmo 1.5 para que resuelva problemas multi-objetivo, requiere un diseño que busca alcanzar tres metas [Padhye, 2009]:

- Maximizar el mayor número de elementos del conjunto de óptimos de Pareto.
- Minimizar la distancia entre las soluciones obtenidas por el algoritmo y el conjunto de óptimos de Pareto.
- Maximizar la dispersión de las soluciones encontradas, de modo que podamos tener una distribución lo más uniforme posible.

La principal dificultad en extender PSO a una versión multi-objetivo, es encontrar la mejor forma de seleccionar al líder de cada partícula en el cúmulo. La dificultad radica en que no hay una noción clara sobre cómo definir cuál es la mejor posición alcanzada por una partícula hasta el momento (***pBest***), así como la mejor posición alcanzada por alguna partícula de la población (***gBest*** o ***lBest*** si se consideran vecindarios) para el caso de optimización multi-objetivo. En los problemas de optimización multi-objetivo, todas las soluciones no dominadas son igualmente buenas, por lo que cualquiera de ellas puede adoptarse como líder. El algoritmo 1.6 muestra una estructura básica de un PSO extendido a su versión multi-objetivo, S es el cúmulo y A^t es el archivo de las mejores soluciones en la generación t.

Existen varias propuestas en la literatura para la selección del líder, tanto para ***gBest*** como para ***pBest***. Las propuestas siguientes son las más populares [Padhye, 2009]:

Algoritmo 1.6 Pseudocódigo del algoritmo PSO multi-objetivo

Entrada: Problema de Optimización.

Salida: Soluciones no dominadas en un archivo acotado A.

1: Iniciar contador de generaciones, $t \leftarrow 0$
2: Iniciar contador ($C \leftarrow 0$) de partículas no dominadas en el archivo A
3: Iniciar aleatoriamente las posiciones $\vec{x}_i^0$ de la población, donde $i = \{0, \ldots, n\}$.
4: Iniciar la velocidad $\vec{v}_i^0 = 0$ de la población, donde $i = \{0, \ldots, n\}$.
5: Si la partícula i no pertenece a la región de búsqueda, se descarta, se muta y se calcula nuevamente.
6: Evaluar las n partículas de la población.
7: Seleccionar el $p\vec{B}est$ inicial de las partículas.
8: Insertar las partículas no dominadas de la población en el archivo A^0.
9: **Mientras** $t < maxG$ **Hacer**
10: Seleccionar un g.
11: Calcular la nueva velocidad y posición de cada partícula en la población.
12: **Para** $i \leftarrow 1$ hasta $size(S)$ **Hacer**
13: Generar una nueva velocidad
$\vec{v}_i^{t+1} \leftarrow \omega \cdot \vec{v}_i^t + \phi_1 \cdot rnd_1 \cdot \left(p\vec{B}est_i^t - \vec{x}_i^k\right) + \phi_2 \cdot rnd_2 \cdot \left(\vec{g}^t - \vec{x}_i^k\right)$
14: Calcular las nuevas posiciones
$\vec{x}_i^{t+1} \leftarrow \vec{x}_i^t + \vec{v}_i^{t+1}$
15: Si la partícula i no pertenece a la región de búsqueda, se descarta, se muta y se calcula nuevamente.
16: Evaluar las n partículas de la población.
17: Actualizar el $p\vec{B}est$ de las partículas.
18: Actualizar las partículas no dominadas en el archivo A^t.
19: $t \leftarrow t + 1$.
20: **Retornar** Soluciones no dominadas en un archivo acotado A.

- **Selección basada en dominancia de Pareto**: Existen tres métodos principales de selección de líderes basados exclusivamente en la dominancia de Pareto para la selección del mejor alcanzado por alguna partícula de la población (***gBest***). Además, estos métodos presentan una menor dependencia del "factor de turbulencia":

  - **ROUNDS**: es un tanto complejo y promueve explícitamente la diversidad en la población mediante la atracción hacia las regiones escasamente pobladas. Los miembros del archivo que dominan la menor cantidad de partículas deben ser asignados preferentemente como líderes globales. Este método puede ser computacionalmente costoso cuando el tamaño del archivo se hace muy grande.
  - **RANDOM**: es simple y promueve la convergencia. Por cada partícula se buscan los miembros en el archivo que lo dominan y se elige uno al azar. Si no existe, se elige aleatoriamente de entre cualquier elemento del archivo.
  - **PROB**: es un método probabilístico ponderado y forma un compromiso entre las dos anteriores. Al igual que en RANDOM, se encuentran todos los miembros del archivo que dominan a cada partícula. Para elegir un líder se le asigna a la partícula una probabilidad de selección inversamente proporcional al número de partículas que la dominan. En el caso de que la partícula no esté dominada por ninguna otra, se selecciona al líder aleatoriamente de entre todo el archivo.

- **Selección basada en Indicadores**: en este método, se seleccionan individuos de acuerdo a su contribución al hipervolumen. El hipervolumen se calcula para todo el archivo y el miembro del archivo cuya contribución del hipervolumen sea mayor es elegido como líder. En el caso de que la partícula no está dominada por ninguno de los miembros del archivo, entonces no hay selección y el líder se toma aleatoriamente del archivo. El método es menos dependiente del factor de turbulencia (uso de un operador de mutación) y funciona muy bien tanto para la selección ***pBest*** como para la del ***gBest*** [Chang et al., 2005].

La literatura presenta varias formas de poder resolver problemas multi-objetivo utilizando un algoritmo de cúmulo de partículas, usando una topología totalmente conectada o con el uso de vecindarios. Por ejemplo, [Parsopoulos and Vrahatis, 2002] usan un enfoque que consiste en combinar todos los objetivos del problema en uno solo, es decir, transforman un problema multi-objetivo en uno mono-objetivo. Otros como [Hu and Eberhart, 2002, Hu et al., 2003] clasifican a los objetivos en orden de importancia, es decir, operan comenzando con el objetivo más importante y luego procesan los objetivos restantes de acuerdo al orden establecido. Existen métodos que hacen uso de varias sub-poblaciones como optimizadores de un solo objetivo. En este caso, las subpoblaciones de alguna manera intercambian diferentes soluciones generadas por las sub-poblaciones que optimizan los objetivos por separado [Mostaghim and Teich, 2004, Martínez and Coello, 2011]. La forma más común de seleccionar al líder es utilizar la dominancia de Pareto, cuya idea básica es seleccionar como líderes a las partículas que son no dominadas con respecto al cúmulo [Coello et al., 2004, Lechuga and Rowe, 2005]. Finalmente, otros autores consideran un enfoque de estrategias combinadas o utilizando una estrategia *maximin* para determinar la dominancia de Pareto [Mostaghim et al., 2007, Fu et al., 2011]. También hay algoritmos que hacen uso de estrategias paralelas utilizando medidas de desempeño [Chang et al., 2005].

La mayoría de estos algoritmos incorporan un esquema dinámico por el peso de la inercia ω, un archivo externo y un factor de turbulencia que es un operador de mutación cuyo propósito es evitar la convergencia prematura a mínimos locales.

1.4. Hipervolumen

El descubrimiento de importantes beneficios del hipervolumen, a comparación de otros indicadores, ha llevado a los investigadores a buscar métodos más eficientes de calcularlo, sobre todo cuando lidiamos con bastantes funciones objetivo. A continuación se hace una descripción del hipervolumen y sus características principales.

1.4.1. Indicador de hipervolumen

El hipervolumen es un indicador muy común que se utiliza para medir y comparar la calidad de las soluciones finales producidas por un algoritmo evolutivo. La métrica del hipervolumen fue propuesta originalmente por [Zitzler and Thiele, 1998], quienes lo definen como "el tamaño del espacio cubierto o del espacio dominado". Hay algunos estudios que analizan el uso de esta medida para la búsqueda multi-objetivo y varios más sobre métodos eficientes para el cálculo del hipervolumen [Fonseca et al., 2006, Beume et al., 2009, Emmerich et al., 2005]

Definición 1.10 (El indicador de hipervolumen)**.** Dada Λ que denota la métrica de *Lebesgue* o métrica $\mathcal{S}$, el hipervolumen es definido formalmente como [Coello et al., 2006]:

$$I_h\left(\mathcal{P}\right) = \mathcal{S}(\mathcal{P}, y_{ref}) = \Lambda\left(\bigcup_{y\in\mathcal{P}}\left\{y' |\ y \prec y' \prec y_{ref}\right\}\right), \mathcal{P} \subseteq \mathbb{R}^m$$

El hipervolumen se define como la medida de *Lebesgue* de la unión de los hipercubos que están limitados por algún punto de referencia y_{ref}, y las imágenes de $y \in \mathcal{P}$ [Zitzler et al., 2006]. Por ejemplo, para hacer el cálculo del hipervolumen para dos funciones objetivo, se define como el área de cobertura del $\mathcal{F}_{conocido}$ en relación con el espacio objetivo de dichas funciones. Esto equivale a la suma de todas las áreas rectangulares, delimitadas por un punto de referencia $\left(f_1\left(x\right), f_2\left(x\right)\right)$ (ver figura 1.13). Matemáticamente, se describe como [Coello et al., 2006]:

$$\mathcal{S}(\mathcal{P}, y_{ref}) = \bigcup_i \left\{vol_i |\ vec_i \in \mathcal{F}_{conocido}\right\}$$

Este indicador tiene dos ventajas importantes, las cuales son [Zitzler et al., 2006]:

1. Es sensible a cualquier tipo de mejora, es decir, cuando se calcula el hipervolumen para un conjunto A de soluciones, que domina a otro conjunto B de soluciones, entonces el aporte del hipervolumen será de mayor calidad para el primer conjunto que para el segundo.

2. Como resultado de la primera propiedad, el hipervolumen garantiza que, cualquier aproximación al conjunto A de soluciones que alcanza el valor máximo de calidad posible para un problema particular contiene todo el conjunto de óptimos de Pareto.

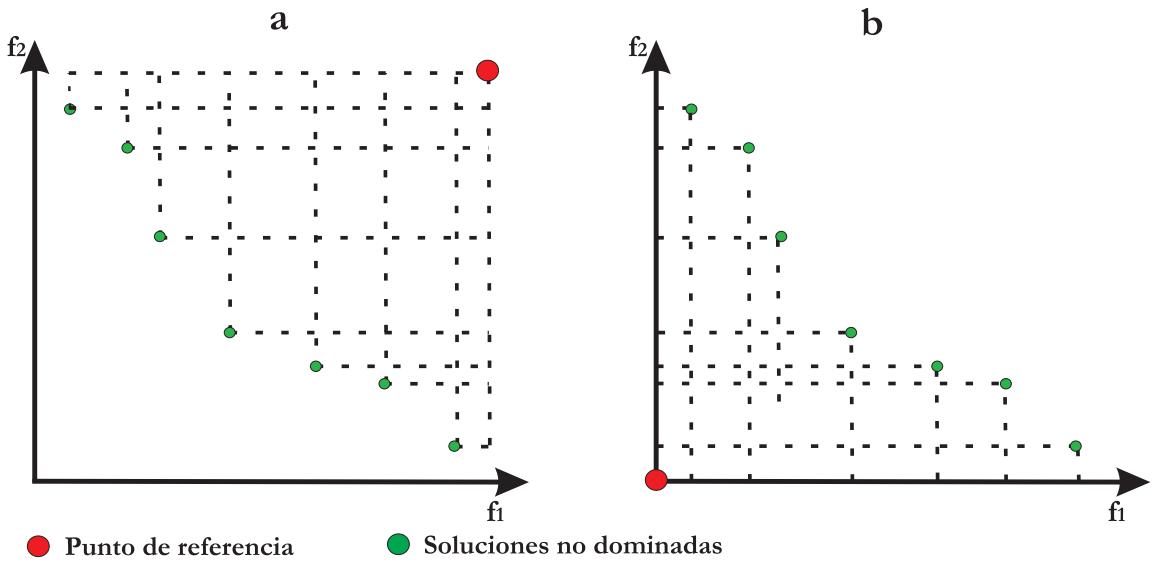

Figura 1.13: (a) El hipervolumen que se centra en un punto de referencia dado. (b) El hipervolumen que se centra en un punto en el origen.

El principal inconveniente de este indicador, es que no existe ningún algoritmo que lo pueda calcular en tiempo polinomial, lo cual, limita un tanto su uso en la práctica. Sin embargo, posee ciertas propiedades que lo hacen muy atractivo [Rodríguez Villalobos, 2011]:

- El hipervolumen es el único indicador de calidad unario, que garantiza la monotonicidad estricta con respecto a la relación de dominancia de Pareto. Un conjunto $\mathcal{P} \in \mathbb{R}^m$ alcanza el valor máximo de hipervolumen si y sólo si todos los puntos de $a \in \mathcal{P}$ son óptimos de Pareto.
- Se puede maximizar o minimizar, pero el punto de referencia debe ser escogido de acuerdo con las siguientes reglas:
  - El punto de referencia debe ser menor o igual que el vector ideal para la minimización del hipervolumen.
  - El punto de referencia debe ser mayor o igual que el vector Nadir para la maximización del hipervolumen.
- Este indicador no es invariante a la escala: es sensible a la elección del punto de referencia. La elección del punto de referencia puede afectar el resultado del hipervolumen (ver figura 1.14) [Reehuis, 2010].
- [Bringmann and Friedrich, 2008] han demostrado que el cálculo del hipervolumen es $\#P$-hard .
- El cálculo del hipervolumen requiere un espacio objetivo normalizado y positivo [Emmerich et al., 2005].
- La pendiente del frente de Pareto determina los puntos que maximizan el hipervolumen.

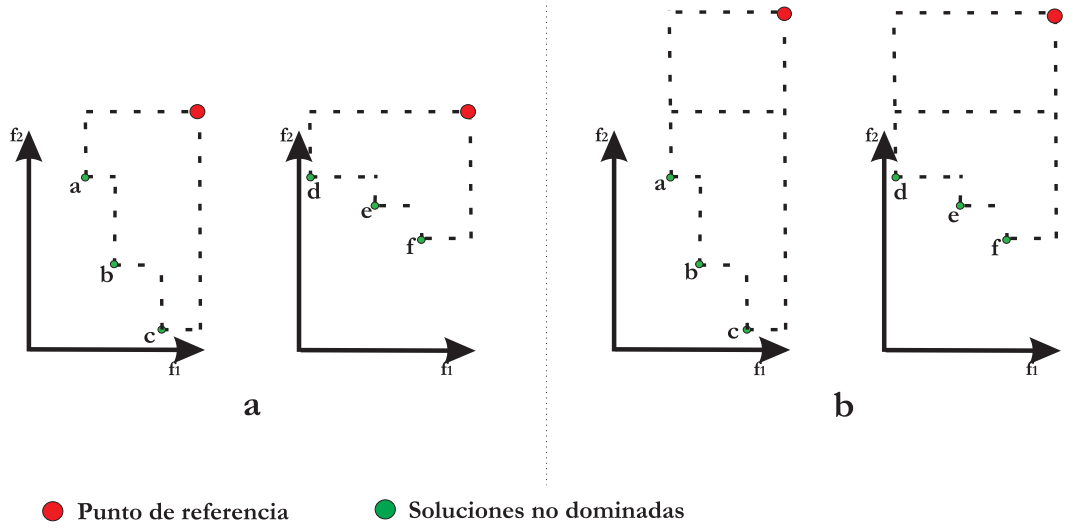

Figura 1.14: El valor relativo del hipervolumen depende de la elección del punto de referencia. Por ejemplo, sean dos conjuntos no dominados. $A = \{1, 2, 3\}$ y $B = \{4, 5, 6\}$, se muestran sus imagenes en el espacio objetivo $\{a, b, c\}$ y $\{d, e, f\}$. En (a) $I_h(A) > I_h(B)$, mientras que en (b) $I_h(A) < I_h(B)$.

Existen diversas propuestas para hacer el cálculo de la métrica del hipervolumen. Por ejemplo: [Beume, 2009] describe la forma de considerar la métrica S como un caso especial de un problema más general llamado problema de la métrica de *Klee*. Para esta métrica existe un algoritmo con tiempo de ejecución $\mathcal{O}(n\log(n)+n^{\frac{d}{2}}\log(n))$, para n puntos con $d \geq 3$ dimensiones. [Fonseca et al., 2006] propoponen un algoritmo con un tiempo de ejecución de $\mathcal{O}(n^{d-2} \cdot \log(n))$.

Otra propuesta la hacen [Yang and Ding, 2007], con un algoritmo que trabaja en un tiempo de ejecución $\mathcal{O}((\frac{d}{2})^n)$ para un conjunto de n puntos incomparables. El algoritmo 1.7 crea un hiper-cubo en d-dimensiones que dividen en secciones o hiper-rectángulos el espacio de los objetivos con algunos puntos de referencia, para posteriormente sumar los valores de las secciones directamente. Este algoritmo requiere de un conjunto de n puntos iniciales de dimensión d obtenidos a lo largo de un hiper-cuboide H y de un conjunto de parámetros, los cuales son: *order*, que es una matriz bidimensional que contiene todos los puntos dominados ordenados por cada dimensión y los vectores *split*, *splitCount* y *coveredCount* que indican los puntos de corte en el espacio y los puntos visitados.

Las entradas del algoritmo son un conjunto de soluciones no dominadas y un punto de referencia, por lo que los hiper-rectángulos se representan de forma implícita. De hecho, cuando el hiper-cuboide se corta en dos hiper-retángulos, puede haber algunos puntos dominados por el punto de referencia en la división cuboide más grande. En este algoritmo no importa si esos puntos se quitan o no.

Por otro lado, el uso de algoritmos que "aproximan" el cálculo de hipervolumen, deben considerar que existe un compromiso entre la precisión y la eficiencia al estimar el valor del hipervolumen en alta dimensionalidad [Braberman et al., 2007, Bader et al., 2008].

Este compromiso es aceptable, siempre y cuando los indicadores que aproximan el valor del hipervolumen, a pesar de la pérdida de precisión adoptada, siguen siendo capaces de clasificar y comparar de forma fiable los conjuntos de soluciones [Bader and Zitzler, 2011]. Se han desarrollado métodos de muestreo (o *Monte Carlo*) para aproximar el hipervolumen

Algoritmo 1.7 $CalcVolume\,(H)$

Entrada: Una población $\mathcal{P} = \{y_1, \ldots, y_n\}$, donde $\mathcal{P}_i = (y_{i1}, \ldots, y_{id})$, un punto de referencia $R = (r_1, \ldots, r_n)$ y $H = \{y_1, \ldots, y_n, r\}$

Salida: El valor del hipervolumen *volume*

Si $n = 1$ **Entonces**
    Retornar $\prod_{j=1}^{d} |r_j - y_{1j}|$
$volume \leftarrow 0$
$splitCount\,[] \Leftarrow n$
Para $j \leftarrow 1, n$ **Hacer**
    ordenar $y_{1j}, \ldots, y_{nj}$
    Para i=1,n **Hacer**
        $order\,[i-1]\,[j-1] \Leftarrow$ número de puntos estrictamente dominados y_{ij}
Para $i \leftarrow 1, n$ **Hacer**
    $coveredCount\,[] \Leftarrow 0$
    Para $j \leftarrow 1, d$ **Hacer**
        $coveredCount\,[order\,[i-1]\,[j-1]] + +$
    Para $k =\leftarrow n-1, 0$ **Hacer**
        Si $coveredCount\,[k] < splitCount\,[k]$ **Entonces**
            $split \leftarrow i$
            $splitCount\,[] \Leftarrow coveredCount\,[]$
            break
Para $j \leftarrow 1, d$ **Hacer**
    Si $order\,[split-1]\,[j-1] > 0$ **Entonces**
        $H2 \Leftarrow \{\}$
        Para $y_i \in H \backslash \{y_{split}, r\}$ **Hacer**
            Si y_{ij} es dominado estrictamente por $y_{split,j}$ **Entonces**
                $H2 \Leftarrow H2 + \{y_i\}$
                $y_{ij} \Leftarrow y_{split,j}$
        $r2 \Leftarrow r$
        $r2\,[j-1] \Leftarrow y_{split,j}$
        $H2 \Leftarrow H2 + \{r2\}$
        $volume \leftarrow volumen + CalcVolume\,(H2)$
$volume \leftarrow volumen + \prod_{j=1}^{d} |r_j - y_{split,j}|$
Retornar *volume*.

el cual ha demostrado ser un buen estimador, capaz de aumentar la precisión conforme se aumenta el número de puntos de la muestra [Everson et al., 2002].

1.4.2. Contribución al hipervolumen

Es un valor que se basa en el indicador del hipervolumen, para obtener la contribución de cada solución al valor total en la maximización del hipervolumen teniendo en cuenta sus vecinos en cada objetivo (ver figura 1.15). El algoritmo 1.8 muestra una forma de calcular la contribución de cada solución usando el algoritmo 1.7 del indicador del hipervolumen, excluyendo una solución a la vez del valor total del hipervolumen. En este algoritmo denominamos con C_a a la contribución al hipervolumen de $a \in \mathcal{P}$. Sin embargo, la selección del conjunto óptimo de μ soluciones implica el cálculo de $\binom{n}{\mu}$ contribuciones al hipervolumen, lo cual es muy costoso en tiempo (computacionalmente hablando).

Algoritmo 1.8 *estimandoContribucion* $(\mathcal{P})$

Entrada: Una población $\mathcal{P} = \{y_1, \ldots, y_n\}$, donde $\mathcal{P}_i = (y_{i,1}, \ldots, y_{n,d})$, un punto de referencia $R = (r_1, \ldots, r_n)$.
Salida: Contribución al hipervolumen $C_{\mathcal{P}}$

1: $C \Leftarrow 0$
2: **Mientras** $\forall a \in \mathcal{P}$ **Hacer**
3: $\quad C_a \leftarrow CalcVolume(\mathcal{P}, r) - CalcVolume(\mathcal{P} \setminus \{a\}, R)$
4: **Retornar** C

En la literatura se pueden encontrar formas más eficientes de hacer el cálculo de la contribución al hipervolumen. Por ejemplo, [Bringmann and Friedrich, 2010] muestran un algoritmo que calcula un conjunto λ de soluciones que menos contribuyen a la maximización del hipervolumen en $\mathcal{O}(n^{\frac{m}{2}} log\ n + n^{\lambda})$, donde n es el número de soluciones, m el número de objetivos y λ el número de soluciones a descartar.

Otro método es descartar de forma iterativa a la población con el menor aporte hasta un tamaño μ. [Bringmann and Friedrich, 2009] demuestran que este método puede dar lugar a un conjunto que no sea el óptimo de acuerdo con la maximización del hipervolumen.

HypE es otro algoritmo basado en muestreo para aproximar las contribuciones al hipervolumen de un conjunto de soluciones $\mathcal{P}$. En HypE se utiliza el método de Monte Carlo, sosteniendo que en los problemas de optimización multi-objetivo, la importancia de los valores de los indicadores reales es relativamente baja en comparación con las soluciones deducidas mediante el uso de hipervolumen [Bader and Zitzler, 2011]. Dada su importancia, HypE se discute a continuación en mayor detalle.

1.4.3. HypE: Estimación de la Contribución del Hipervolumen

El algoritmo HypE aborda la cuestión de cómo calcular los valores de contribución al hipervolumen para una población $\mathcal{P} \in \psi$. [Bader and Zitzler, 2011] proponen determinar los valores de contribución de a, denotada como $C_h^k(a, \mathcal{P}, \mathcal{R})$, para todos los elementos de $a \in \mathcal{P}$ con un punto fijo de referencia $\mathcal{R}$ de manera exacta (ver algoritmo 1.9). Este opera de acuerdo al principio de "hipervolumen por corte de objetivos", el cual difiere de los métodos existentes permitiendo: (i) considerar un conjunto de puntos de referencia $\mathcal{R}$ y (ii) obtener todos los valores de contribución de la población $\mathcal{P}$. La complejidad de tiempo de ejecución

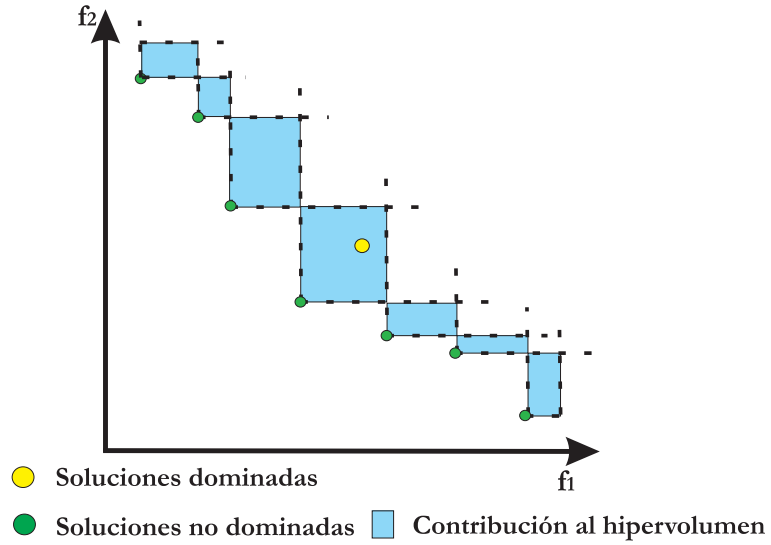

Figura 1.15: Contribución al Hipervolumen

del peor caso es $\mathcal{O}(|\mathcal{P}|^d + d|\mathcal{P}| \log |\mathcal{P}|)$, suponiendo que la población ya está ordenada en todas las dimensiones, denotando a d como la dimensión de los objetivos.

El algoritmo 1.9 corta el espacio dominado de forma recusiva, devolviendo un valor de aptitud para cada $a \in \mathcal{P}$ correspondiente al par (a, v) donde v es el valor de contribución de a. En cada nivel de recursión, se hace una exploración a lo largo de un objetivo en específico, dado i con u^* que representa la posición actual de exploración. El vector $(z_1, \ldots, z_d)$ contiene todas las posiciones de las dimensiones de la exploración y en cada invocación los vectores objetivo y los puntos de referencia son filtrados de acuerdo con estas posiciones, y también se pueden seleccionar las soluciones dominadas. Además, el volumen parcial V se actualiza antes de invocarlo recursivamente basado en la distancia a la posición siguiente de exploración, por lo que en el nivel de recursión más bajo ($i = 0$), la variable V contiene el hipervolumen de todos los objetivos. Se tiene en cuenta que la población es un conjunto múltiple, es decir, puede contener soluciones duplicadas. Por lo tanto, todos los conjuntos en el algoritmo son múltiples, donde k es el número de soluciones a descartar (comúnmente $k = 1$).

Debido a que el algoritmo 1.9 es viable sólo para un número reducido de objetivos, se utiliza el método de Monte Carlo que es conocido por ser fácil de usar para resolver problemas numéricamente mediante el uso de números aleatorios. Dado que una de las aplicaciones más comunes del método de Monte Carlo es el cálculo de integrales, éste es una buena opción para estimar el hipervolumen. El algoritmo 1.10 muestra una forma de aproximar el valor de la contribución al hipervolumen para una población $\mathcal{P}$.

Para este propósito, se define un espacio de muestreo $S \subseteq Z$. Las muestras se toman dentro de una caja delimitada por un conjunto de puntos, es decir:

$$S = \{(z_1, \ldots, z_d)\} \in Z |\ 1 \leq i \leq d : l_i \leq z_i \leq u_i$$

donde $l_i = \overset{\text{mín}}{a \in \mathcal{P}} f_i(a)$ y $u_i = \overset{\text{máx}}{(r_1,\ldots,r_d) \in \mathcal{R}} r_i$, para $1 \leq i \leq d$, por lo que, el volumen V del espacio de muestreo de S está definido por $V = \prod_{i=1}^{d} \text{máx}\{0, u_i - l_i\}$.

Algoritmo 1.9 *hypeExact* $(\mathcal{P}, R, k, i, V, (z_1, \ldots, z_d))$. Cálculo exacto de la contribución del Hipervolumen

Entrada: $\mathcal{P} = \{y_1, \ldots, y_n\}$, donde $\mathcal{P}_i = (y_{i,1}, \ldots, y_{n,d})$, $\Gamma = \bigcup_{a \in \mathcal{P}} \{(a, 0)\}$, un punto de referencia $R = (r_1, \ldots, r_n)$, un parámetro $k \in \mathbb{N}$

Salida: Contribución del hipervolumen $C_{\mathcal{P}}$

Se filtran las soluciones relevantes $A_U \leftarrow \bigcup_{(a,v) \in \Gamma, \forall i < j \leq d : f_j(a) \leq z_j} \{f(a)\}$
Se filtran los puntos de referencia $U_R \leftarrow \bigcup_{(r_1,\ldots,r_d) \in R, \forall i < j \leq d : r_j \geq z_j} \{(r_1, \ldots, r_d)\}$
Si $i = 0 \wedge U_R \,! = 0$ **Entonces**
    $\alpha \leftarrow \prod_{j=1}^{|A_U|-1} (k - j) \,/\, (|\Gamma| - j)$.
    $\Gamma^{'} \leftarrow 0$
    Para $(a, v) \in \Gamma$ **Hacer**
        Si $\forall 1 \leq j \leq d : f_j(a) \leq z_j$ **Entonces**
            $\Gamma^{'} \leftarrow \Gamma^{'} \cup \left\{ \left(a, v + \frac{\alpha}{|A_u|} \cdot V \right) \right\}$
        Sino
            Se actualiza el hipervolumen de las soluciones filtradas
            $\Gamma^{'} \leftarrow \Gamma^{'} \cup \{(a, v)\}$
Sino
    Si $i > 0$ **Entonces**
        $\Gamma^{'} \leftarrow \Gamma$
        $U \leftarrow A_U \cup U_R$
        Continua la recursión
        Mientras $U^{'} \,! = \emptyset$ **Hacer**
            $u^* \leftarrow min_{(u_1,\ldots,u_d) \in U} u_i$
            $U^{'} \leftarrow \{(u_1, \ldots, u_d) \in U | u_i > u^*\}$
            Se analiza la dimensión actual en forma ascendente
            Si $U' \,! = \emptyset$ **Entonces**
                $V^{'} = V \cdot \left(\left(min_{(u'_1,\ldots,u'_d) \in U'} u_i^{'} \right) - u^* \right)$
                $\Gamma^{'} \leftarrow hypeExact \left(\Gamma^{'}, R, k, i - 1, V^{'}, (z_1, \ldots, z_{i-1}, u^*, z_{i+1}, \ldots, z_d) \right)$
            $U \leftarrow U^{'}$
$C \leftarrow \Gamma^{'}$
Retornar C

Algoritmo 1.10 *hypeSampling* $(\mathcal{P}, R, k, i, V, (z_1, \ldots, z_d), M)$. Estimando el cálculo de la contribución del Hipervolumen

Entrada: $\mathcal{P} = \{y_1, \ldots, y_n\}$, donde $\mathcal{P}_i = (y_{i,1}, \ldots, y_{n,d})$, un punto de referencia $R = (r_1, \ldots, r_n)$, un parámetro $k \in \mathbb{N}$ y un número de muestras $M \in \mathbb{N}$

Para $i \leftarrow 1, n$ **Hacer**
  $l_i \leftarrow min_{a \in \mathcal{P}} f_i(a)$
  $U_i \leftarrow max_{(r_1, \ldots, r_d) \in \mathcal{R}} r_i$
$S \leftarrow [l_1, u_1] \times \ldots \times [l_d, u_d]$
$V \leftarrow \prod_{i=1}^{d} max\{0, (u_i, l_i)\}$
$\Gamma \leftarrow \bigcup_{a \in \mathcal{P}} \{(a, 0)\}$
Para $j \leftarrow 1, m$ **Hacer**
  elegir $s \in S$ aleatoriamente
  Si $\exists r \in \mathcal{R} : s \leq r$ **Entonces**
    $A_U \leftarrow \bigcup_{a \in \mathcal{P}, f(a) \leq s} \{f(a)\}$
    Si $|A_U| \leq k$ **Entonces**
      $\alpha \leftarrow \prod_{l=1}^{|A_U|-1} \frac{k-l}{|\mathcal{P}|-l}$
      $\Gamma' \leftarrow 0$
      Para $(A, V) \in F$ **Hacer**
        Si $f(a) \leq s$ **Entonces**
          $\Gamma' \leftarrow \Gamma' \cup \left\{ \left(a, v + \frac{\alpha}{|A_U|} \cdot \frac{V}{m} \right) \right\}$
        Sino
          $\Gamma' \leftarrow \Gamma' \cup \{(a, v)\}$
      $\Gamma' \leftarrow \Gamma$
$C = \Gamma'$
Retornar C

La manera de hacerlo es seleccionar al azar de manera uniforme varios objetivos $(s_1, \ldots, s_m)$ de la caja de muestreo S. Para cada s_j se comprueba si existe dentro de algún par $I_i\,(a, \mathcal{P}, \mathcal{R})$, con $1 \leq i \leq k$. Esto se hace en dos pasos: en primer lugar, se verifica que s_j esté por debajo del conjunto de referencia $\mathcal{R}$, es decir, si existe $r \in R$, que esté dominado por s_j; en segundo lugar, se verifica que en el conjunto $\mathcal{P}$ exista algún miembro que domine a s_j. Si ambas condiciones se cumplen, entonces el punto de muestreo s_j se encuentra en todas las partes $I_i\,(a, \mathcal{P}, \mathcal{R})$ donde $i = |\mathcal{P}|$ y $a \in \mathcal{P}$. Esta situación se denota como un acierto del par i de $a \in \mathcal{P}$ y si alguna de las dos condiciones anteriores no se cumplen, se considera un fallo. Denotan a $X_j^{(i,a)}$ como una variable aleatoria que toma valor igual a 1 en el caso de un acierto de s_j con respecto al par i y 0 en caso contrario. Basándose en los m puntos de muestreo, se estima el valor de $C_i\,(a, \mathcal{P}, \mathcal{R})$ simplemente contando el número de aciertos y multiplicándlo por el volumen de la caja de muestreo, lo que se expresa de la siguiente manera:

$$I_i\,(a, \mathcal{P}, \mathcal{R}) = \frac{\sum_{j=1}^{m} X^{(i,a)}}{m} \cdot V$$

De alguna manera esta técnica aprovecha el valor exacto del hipervolumen, incrementando m por un número muy grande. Por lo que, los autores proponen utilizar la siguiente fórmula para estimar la contribuciones:

$$C_i\,(a, \mathcal{P}, \mathcal{R}) = \sum_{i=1}^{k} \frac{\alpha_i}{i} \left(\frac{\sum_{j=1}^{m} X^{(i,a)}}{m} \cdot V \right)$$

donde $\alpha_i = \prod_{l=1}^{i-1} \frac{k-l}{|\mathcal{P}|-l}$.

Capítulo 2

Algoritmo MOPSO-hv

En este cápitulo se describe el algoritmo de cúmulo de partículas multi-objetivo propuesto en este trabajo de tesis. Dicho algoritmo tiene un mecanismo de selección basado en la contribución al hipervolumen y adopta un operador de turbulencia. El algoritmo propuesto, se denominó *multi-objective particle swarm optimizer based on hypervolume (MOPSOhv)*.

Descripción del algoritmo

La idea principal de nuestro algoritmo es mantener un cúmulo de partículas, cuyas soluciones sean las mejores posibles de todo el espacio de búsqueda. En éste, se establece un vecindario global ($gBest$) del cual se escoge (desde un archivo que contiene las mejores soluciones encontradas hasta el momento), como la mejor, a la partícula que contribuye más al valor del hipervolumen, a fin de favorecer la explotación del espacio de soluciones. Para beneficiar la exploración en el espacio de búsqueda, se utiliza un operador de turbulencia el cual también busca prevenir la convergencia prematura, debido a que algunos problemas de optimización multi-objetivo son multi-frontales. Siguiendo el algoritmo 1.6 (ver página 27) se tiene el algoritmo 2.1, utilizando la siguiente notación:

- tam_S: es el tamaño de la población principal.
- tam_A: es el tamaño de la población secundaria o del archivo A.
- dim: es e número de objetivos del problema.
- max_{Gen}: es el número máximo de generaciones.
- t: es el contador del número de iteraciones o generaciones.
- n_d: es el contador de partículas no dominadas en el archivo.
- S: contiene a la población o cúmulo principal, donde $S = s_{i,k}$ con $i = 1, \ldots, tam_S$ y $k = 1, \ldots, dim$.
- A: contiene a la población secundaria o del archivo, donde $A = a_{j,k}$ con $j = 1, \ldots, tam_A$ y $k = 1, \ldots, dim$.
- $\vec{l}$, $\vec{u}$: son los intervalos inferior y superior del problema de optimización multi-objetivo, donde $\vec{l} = \{l_1, \cdots, l_{dim}\}$ y $\vec{u} = \{u_1, \cdots, u_{dim}\}$.

- v: contiene la velocidad de todas las partículas en el cúmulo, donde $v = v_{i,k}$ con $i = 1, \ldots, tam_S$ y $k = 1, \ldots, dim$.
- p: contiene el valor objetivo de todas las partículas en el cúmulo, donde $p = p_{i,k}$ con $i = 1, \ldots, tam_S$ y $k = 1, \ldots, dim$.
- $\vec{w}$: contiene a la partícula que menos contribuye al valor hipervolumen de la población secundaria, $\vec{w} = \{w_1, \cdots, w_{tam_S}\}$.
- F: es el problema de optimización multi-objetivo, donde $F = \{f_1, \cdots, f_k\}$ con $k = 1, \cdots, dim$.
- $\vec{C}$: contiene el valor de la contribución al hipervolumen de todas las partículas del cúmulo S, donde $\vec{C} = \{C_1, \cdots, C_{tam_S}\}$.
- $\vec{ref}$: contiene el punto de referencia para hacer el cálculo del hipervolumen, donde $\vec{ref} = \{ref_1, \cdots, ref_{dim}\}$.

Algoritmo 2.1 Algoritmo PSO multi-objetivo basado en Hipervolumen

Entrada: Problema de Optimización.
Salida: Soluciones no dominadas en un archivo acotado A.

1: Inicializar el contador de generaciones con $t = 0$.
2: Inicializar aleatoriamente las posiciones y la velocidad del cúmulo S.
3: Reintegrar las partículas que no pertenezcan a la región de búsqueda.
4: Evaluar las partículas del cúmulo S.
5: Seleccionar el $pBest$ inicial de las partículas.
6: Insertar las partículas no dominadas de la población en el archivo A.
7: **Mientras** $t < max_{Gen}$ **Hacer**
8: Calcular las contribuciones al hipervolumen del cúmulo S
Calcular la nueva velocidad y posición de cada partícula en la población.
9: **Para** $i = 1$ hasta tam_S **Hacer**
10: Seleccionar un $gBest$ conforme a una porción de los mejores (por ejemplo 5 %).
Se ordena el archivo de manera descendiente de acuerdo a su contribución al hipervolumen.
11: Generar una nueva velocidad
$\vec{v}_i^{t+1} = \omega \cdot \vec{v}_i^t + \phi_1 \cdot rnd_1 \cdot \left(p\vec{B}est_i^t - \vec{S}_i^t\right) + \phi_2 \cdot rnd_2 \cdot \left(g\vec{B}est - \vec{S}_i^t\right)$
12: Calcular las nuevas posiciones
$\vec{S}_i^{t+1} = \vec{S}_i^t + \vec{v}_i^{t+1}$
13: Reintegrar las partículas que no pertenezcan a la región de búsqueda.
14: Aplicar el operador de mutación.
15: Evaluar las partículas del cúmulo S.
16: Actualizar el $pBest$ de las partículas.
17: Actualizar las partículas no dominadas en el archivo A.
18: $t = t + 1$.
19: **Retornar** A.

A continuación se hace una descripción más detallada del algoritmo 2.1:

1. Inicializar la población:

   - Inicializar el contador de generaciones con $t = 0$.
   - Inicializar el contador de partículas no dominadas con $nd = 0$
   - Inicializar el punto de referencia:

   $$ref^t = (0_1, 0_2, \ldots, 0_k)$$

   - Inicializar los valores de la población de manera aleatoria. Todas las partículas i en cada objetivo j, se inicializan con un valor aleatorio en el vecindario determinado por l y u.

   $$S^t_{i,k} = Random(l, u)$$

   - Calcular velocidad inicial. Todas las partículas i y en todos los objetivos k, inicializan su velocidad en cero.

   $$v^t_{i,k} = 0$$

   - Evaluar todas las partículas de la población inicial S^t. Se evaluán o se obtienen los valores objetivo de todas las partículas i en cada objetivo k.

   $$p^t_{i,k} = F\left(S^t_{i,k}\right)$$

   - Almacenar el $pBest$ inicial de todas las partículas del cúmulo. Se copian todas las partículas i del cúmulo en el $pBest$ inicial.

   $$pBest^t = S^t$$

2. Almacenar las soluciones no dominadas encontradas en S^t en el archivo A.

   - Determinar la factibilidad y la no dominacia de una partícula i.
   - Si el archivo esta vac]ío, se almacena la partícula i. De lo contrario, se verifica si la partícula i es factible y si no es dominada por alguna otra partícula ya almacenada en A. Todas las soluciones dominadas por la nueva solución son eliminadas.
   - Si la partícula es factible y no es dominada por alguna otra, entonces ésta se almacena.

3. Actualizar $gBest$.

   - Actualizar el punto de referencia como:

   $$ref^t = \left(\overset{\text{máx}}{i \in S} S_1, \ldots, \overset{\text{máx}}{} S_{dim}\right) + \delta$$

   Si se encuentra un valor peor que ref^{t-1} se actualiza el punto de referencia.
   - Calcular la contribución al hipervolumen de cada partícula i en el archivo A: Para hacer el cálculo de la contribución al hipervolumen, se utiliza el algoritmo 1.8 (ver página 33) que hace uso de la métrica del hipervolumen (ver figura 2.1).

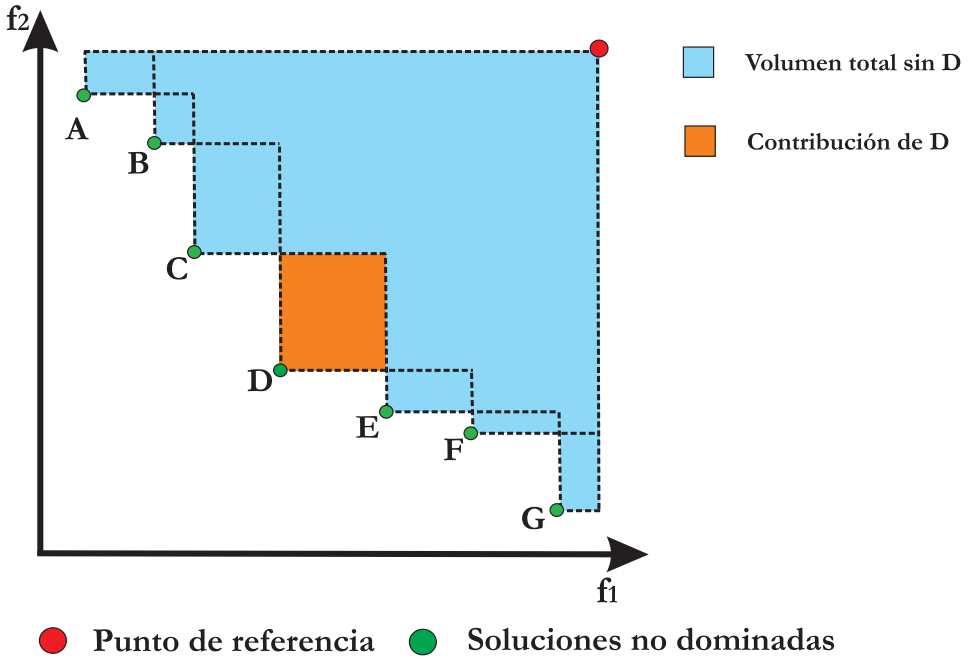

Figura 2.1: Ejemplo para calcular la contribución para D utilizando el algoritmo 1.8. La contribucón de D es igual a la diferencia entre el volumen total calculado con D y el volumen sin D.

Se hace notar, sin embargo, que este algoritmo es muy costoso en tiempo, por lo que se decidió mejor hacer uso del algoritmo de HypE para hacer el cálculo de la contribución al hipervolumen (figuras 2.2, 2.3 y 2.4).

Para realizar la contrinución al hypervolumen se utiliza el algoritmo 2.2: Si $dim < 3$, entonces, se hace el cálculo de la contribución al hipervolumen utilizando el algoritmo (exacto) 1.9 (ver página 35). De lo contrario, se hace uso del algoritmo 1.10 (ver página 36), que aproxima la contribución al hipervolumen (con un número de muestras igual a $m = 10000$), usando cero como intervalo inferior y el peor valor del punto de referencia como intervalo superior.

Algoritmo 2.2 Calcular la contribución de las partículas de la población

1: **Si** $dim < 3$ **Entonces**
2: $hypeExact(C, tam_A, ref, 1, S)$
3: **Sino**
4: $m = 10000$
5: $hypeSampling(C, tam_A, 0, ^{\text{máx}}\{ref\}, m, 1, S)$
6: **Retornar** C.

- Ordenar a la población en el archivo A de manera descendiente conforme a su contribución al hipervolumen.
- Encontrar la partícula w, cuyo valor es el que menos contribuye al hipervolumen.

a) Para cada partícula i del cúmulo S, se hace lo siguiente:

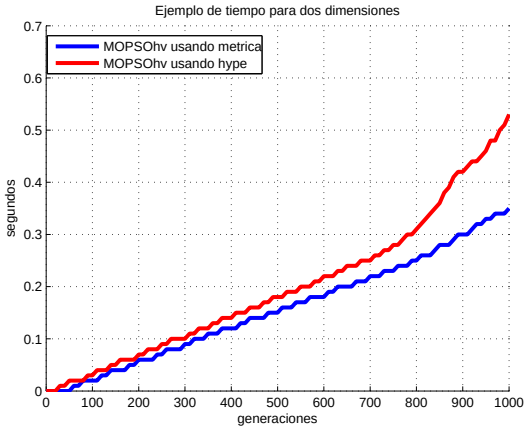

Figura 2.2: Tiempo requerido en dos objetivos por el algoritmo 2.1 usando la métrica completa y el algoritmo Hype en las primeras mil generaciones.

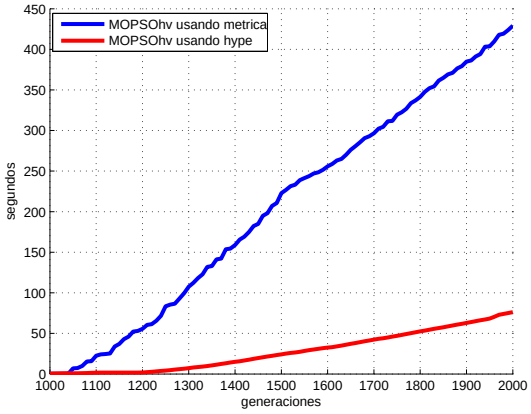

Figura 2.3: Tiempo requerido en dos objetivos por el algoritmo 2.1 usando la métrica completa y el algoritmo Hype después de mil generaciones.

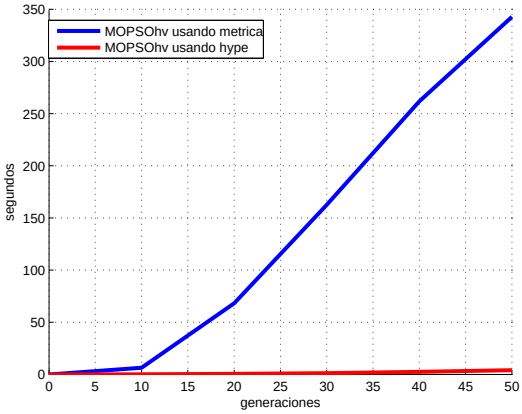

Figura 2.4: Tiempo requerido en tres objetivos por el algoritmo 2.1 usando la métrica completa y el algoritmo Hype en las primeras 50 generaciones.

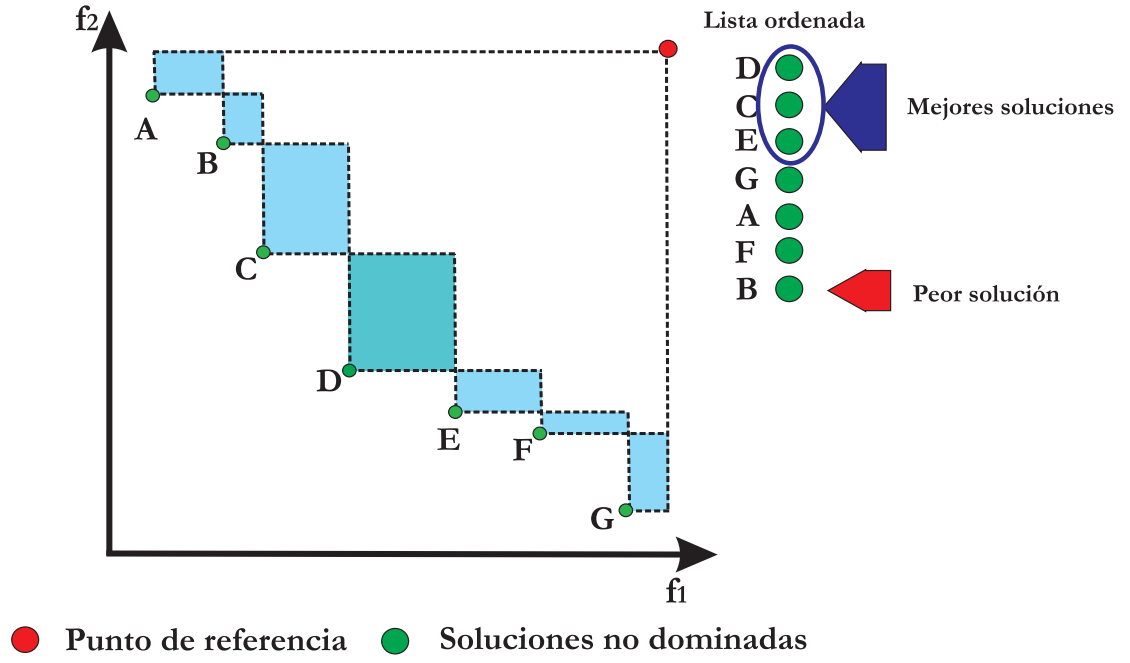

Figura 2.5: Un conjuto de soluciones no dominadas ordenadas de manera descendente conforme a su contribución al hipervolumen.

Actualizar la velocidad y la posición.

- Seleccionar aleatoriamente el mejor guía global ($gBest$) para la partícula $\mathcal{P}_i$ de un porcentaje p de las mejores partículas de la población secundaria, la cuál es ordenada de manera descendente confome a su contribución al hipervolumen (en nuestro caso, $p = 0.01$ del porcentaje de la población secundaria actual). En la figura 2.5 se muestra un ejemplo de cómo ordenar la población del archivo de manera descendente conforme a su contribución al hipervolumen.
- Actualizar velocidad.
  Se utiliza la ecuación (1.2) (ver página 20) para actualizar la velocidad de cada partícula i del cúmulo S. Para acelerar la convergencia se hace uso de un factor de inercia $\omega = 0.4$, por tanto, la ecuación utilizada es la (1.6) (ver página 25). Finalmente, la ecuación queda de la siguiente manera:

$$v_{i,k}^{t+1} = \omega \cdot v_{i,k}^{t} + \phi_1 \cdot rnd_1 \cdot \left(pBest_i^t - S_{i,k}^t\right) + \phi_2 \cdot rnd_2 \cdot \left(gBest - S_{i,k}^t\right),$$

  donde $\phi_1 = \phi_2 = 1$, $rnd_1 = Random(0,1)$ y $rnd_2 = Random(0,1)$. Cada partícula es evaluada en cada objetivo k.
- Actualizar posición.
  Se utiliza la ecuación (1.1) (ver página 20) para actualizar la posición de cada partícula i en cada objetivo k. La ecuación queda de la siguiente manera:

$$S_{i,k}^{t+1} = S_{i,k}^{t} + v_{i,k}^{t+1}$$

- Si la partícula i se sale de la región de búsqueda, entonces, hay que reintegrarla utilizando los límites de búsqueda correspondientes al problema. Se multiplica la velocidad de la partícula i por -1 para cambiar la dirección de búsqueda.
- La mutación es una parte importante para favorecer la explotación del espacio de búsqueda. Se aplica el factor de turbulencia si se cumple que t es menor a $max_{Gen} \cdot pMuta$.
  El algoritmo de cúmulos de partículas tiene una alta velocidad de convergencia. Sin embargo, esta elevada velocidad de convergencia puede ser perjudicial en el contexto de optimización multi-objetivo, debido a que puede convergerse a un falso frente de Pareto (es decir, el equivalente de un óptimo local en la optimización global). Para ello, introdujimos el operador de mutación que se muentra en el algoritmo 2.3 [Coello et al., 2004].

Algoritmo 2.3 Operador de Mutación

Entrada: Partícula S_i para mutar, dim el número de dimensiones, max_{Gen} el número máximo de generaciones, $pMuta$ la probabilidad de mutación y t la generación actual.
Salida: Partícula S_i mutada.

1: **Si** $flip\left((1 - t/max_{Gen})^{5/pMuta}\right)$ **Entonces**
2: $wdim = random(0, dim - 1)$
3: $rango = (u_{wdim} - l_{wdim}) \cdot (1 - t/max_{Gen})^{5/pMuta}$
4: $ub = S_{i,wdim} + rango$
5: $lb = S_{i,wdim} + rango$
6: **Si** $lb < l_{wdim}$ **Entonces**
7: $lb = l_{wdim}$
8: **Si** $ub > u_{wdim}$ **Entonces**
9: $ub = u_{wdim}$
10: $S_{i,wdim} = random\,(lb, ub)$
11: **Retornar** S_i

b) Evaluar todas las partículas de la población S^t. Se evalúan los valores de todas las partículas i en cada función objetivo k.

$$p^t_{i,k} = F\left(S^t_{i,k}\right)$$

c) Actualizar las soluciones no dominadas en A^t.

- Se insertan todas las soluciones no dominadas de S^t en A^t, sólo si las soluciones no son dominadas por alguna ya almacenada. Todas las soluciones dominadas por la nueva solución son eliminadas del archivo.
- Si el archivo A ya está lleno, es decir, $nd > tam_A$. Las soluciones son reemplazadas conforme al siguiente criterio: se hace uso del algoritmo 1.10 que aproxima la contribución (de la misma forma que el paso 3). Se reemplaza la partícula w, que es la que menos contribuye, con la nueva partícula. Se calcula la contribución y se encuentra otra vez a la partícula w.

d) Actualizar el $pBest$ de cada partícula i en cada dimensión k.

Una forma de realizarlo es si el actual $pBest_i$ es mejor que el almacenado en memoria, entonces se actualiza el $pBest^t$, de la siguiente manera:

$$pBest^t_{i,k} = S^t_{i,k}$$

Siguiendo esta topología no se tiene una buena interacción con las demás partículas para generar buenas soluciones. Por lo tanto, se decidió actualizar el $pBest^t$ utilizando la población secundaria (sin considear el porcentaje p de las mejores partículas de esta población), es decir, solo se comparte información de un porcentaje r de partículas que pertenecen a las mejores soluciones generadas hasta el momento, las cuales serán utlizadas en el entorno social del algoritmo (para nuestro caso $r = 1$). El $pBest^t$ se actualiza de acuerdo al algoritmo 2.4.

Algoritmo 2.4 Actualizar $pBest^t$

1: $nttop = ((tam_A - 1) \cdot r)$
2: $top = ((tam_A - 1) \cdot p)$
3: **Para** cada partícula i hasta tam_S **Hacer**
4: $\quad j = RandomInt(top + 1, nttop)$
5: $\quad pBest^t_i = A^t_j$
6: **Retornar** $pBest^t$.

e) Incrementar el contador de generaciones en uno.

$$t = t + 1$$

f) Si no se cumple el criterio de paro, retornar al paso 3.

4. Retornar al cúmulo con la aproximación al frente de Pareto.

Capítulo 3

Evaluación del algoritmo propuesto y resultados

En este capítulo se presenta la evaluación experimental del algoritmo desarrollado en este trabajo de tesis. La evaluación se realizó utilizando un conjunto diverso de problemas multi-objetivo que reunen diferentes características que causan dificultades a un algoritmo evolutivo multi-objetivo. El estudio experimental descrito en este capítulo está dividido en tres partes: la primera parte muestra los resultados para el conjunto de problemas ZDT (*Zitzler-Deb-Thiele*), la segunda muestra los resultados para el conjunto de problemas de DTLZ (*deb-Thiele-Laumanns-Zitzler*). Finalmente, la tercera y última parte muestra los resultados de escalabilidad. Los resultados se contrastan con las soluciones obtenidas por los algoritmos NSGA-II (*the Nondominated Sorting Genetic Algoritm -II*), SMS-EMOA (*Multi-objective Selection based on Dominated Hypervolume*) y MOPSOcd (*Multi-objective Particle Swarm Optimizer based on Crowding Distance*).

Parámetros

Los parámetros utilizados para generar las soluciones de los algoritmos MOPSOhv y MOPSOcd se muentran en la tabla 3.1. Se usó un factor de inercia $\omega = 0.4$ y constantes sociales $\varphi_1 = \varphi_2 = 1$.

	Generaciones	**Población**	**Mutación**
MOPSOhv	1200	100	0.5
MOPSOcd	1200	100	0.5

Tabla 3.1: Parámetros para los algoritmos de cúmulos de partículas

Los parámetros utilizados para generar las soluciones del NSGA-II se muestran en la tabla 3.2. Los parámetros utilizados para generar las soluciones del SMS-EMOA se muestran en la tabla 3.3.

Generaciones	1200
Población	100
Cruza	0.9
Mutación	$\frac{1}{\vec{x}}$

Tabla 3.2: Parámetros de NSGA-II

Generaciones	1200
Población	100
$\eta_c = \eta_m$	20
Mutación	$\frac{1}{\vec{x}}$

Tabla 3.3: Parámetros de SMS-EMOA

El código está implementado en lenguaje C y la evaluación del conjunto diverso de problemas multi-objetivo se realizaron en una computadora portátil con las siguientes características: un procesador Intel(R) Core(TM)2 Duo CPU T5800 a 2.00GHz, 4 Gb de Memoria RAM, un disco duro de 320 Gb y un sistema operativo Ubuntu 12.04.1 LTS de 64 bits.

El uso de una sola métrica difícilmente puede reflejar el desempeño global de un algoritmo evolutivo multi-objetivo. Algunas métricas miden la convergencia del algoritmo al frente de Pareto real, otras la distribución y la diversidad de las soluciones. En este caso necesitamos evaluarlo considerando varias métricas simultáneamente. Sin embargo, la métricas evaluán dos objetivos en conflicto (la convergencia y la diversidad). Si los valores de las métricas de un algoritmo dominan a las de otro, entonces podemos afirmar que el primero es mejor que el segundo. En otro caso, no podemos concluir nada acerca de los dos algoritmos. Las métricas que se utilizarán para evaluar la eficacia son el espaciamiento para evaluar la distribución; y la distancia generacional invertida, la cobertura de dos conjuntos y el indicador del hipervolumen para evaluar la convergencia. La definición formal de las métricas utilizadas se muestran en la sección 1.2.1 en la página 9.

3.1. Conjunto de problemas ZDT

Estos problemas están diseñados poniendo 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, se describe a continuación [Zitzler et al., 2000]:

$$\begin{aligned} \text{Minimizar} \quad & F = (f_1(x_1), f_2(x_2)) \\ \text{Sujeto a} \quad & f_2(x) = g(x_2, \ldots, x_m) \cdot h(f_1(x_1), g(x_2, \ldots, x_m)), \\ \text{donde} \quad & x = (x_1, \ldots, x_m) \end{aligned}$$

La funcion 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:

$$\begin{aligned} f_1(x) &= x_1 \\ f_2(x, g(x)) &= g(x) \cdot \left(1 - \sqrt{\frac{f_1}{g(x)}}\right) \\ g(x) &= 1 + \frac{9}{n-1} \cdot \sum_{i=2}^{n} x_i \end{aligned}$$

  donde $n = 30$ y $x_i \in [0, 1]$ con $i = 1, \ldots, n$. El frente de Pareto real se forma con $g(x) = 1$ (figura 3.1).

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

$$\begin{aligned} f_1(x) &= x_1 \\ f_2(x, g(x)) &= g(x) \cdot (1 - \left(\frac{f_1}{g(x)}\right)^2) \\ g(x) &= 1 + \frac{9}{n-1} \cdot \sum_{i=2}^{n} x_i \end{aligned}$$

  donde $n = 30$ y $x_i \in [0, 1]$ con $i = 1, \ldots, n$. El frente de Pareto real se forma con $g(x) = 1$ (figura 3.2).

- La función de prueba **ZDT3** tiene un frente de Pareto discontinuo y convexo:

$$\begin{aligned} f_1(x) &= x_1, \\ f_2(x, g) &= g(x) \cdot \left(1 - \sqrt{\frac{f_1(x)}{g(x)}} - \frac{f_1(x)}{g(x)} \operatorname{sen}(10 \cdot \pi \cdot f_1(x))\right) \\ g(x) &= 1 + \frac{9}{n-1} \cdot \sum_{i=2}^{n} x_i \end{aligned}$$

donde $n = 30$ y $x_i \in [0, 1]$ con $i = 1, \ldots, n$. El frente de Pareto real 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 3.3).

- 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:

$$
\begin{aligned}
f_1(x) &= x_1, \\
f_2(x, g(x)) &= g(x) \cdot \left(1 - \sqrt{\frac{f_1(x)}{g(x)}}\right), \\
g(x) &= 1 + 10 \cdot (n-1) + \sum_{i=2}^{n} (x_i^2 - 10 \cdot \cos(4 \cdot \pi \cdot x_i))
\end{aligned}
$$

donde $n = 10$, $x_1 \in [0, 1]$ y $x_i \in [-5, 5]$ con $i = 2, \ldots, n$. El frente de Pareto real se forma con $g(x) = 1$ (figura 3.4).

- La función de prueba **ZDT6** tiene un espacio de búsqueda no uniforme:

$$
\begin{aligned}
f_1(x) &= 1 - e^{(-4 \cdot x_1)} \cdot \text{sen}^6(6 \cdot \pi \cdot x_1), \\
f_2(x, g(x)) &= g(x) \cdot (1 - \left(\frac{f_1}{g(x)}\right)^2), \\
g(x) &= 1 + 9 \cdot \left[\frac{\sum_{i=2}^{n}}{9}\right]^{0.25}
\end{aligned}
$$

donde $n = 10$, $x_1 \in [0, 1]$ y $x_i \in [-5, 5]$ con $i = 2, \ldots, n$. El frente de Pareto real se forma con $g(x) = 1$. Este problema tiene una baja densidad en las soluciones cerca del frente de Pareto real y una alta densidad lejos del mismo (figura 3.5).

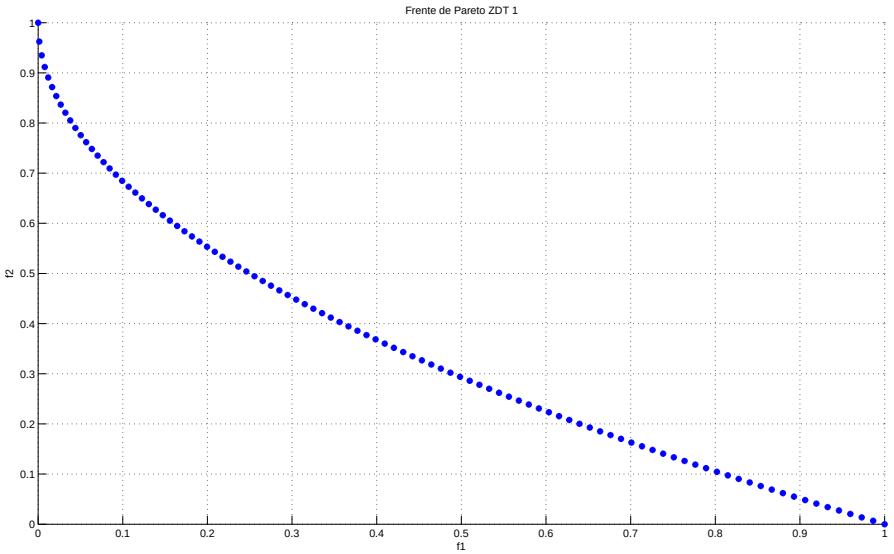

Figura 3.1: Frente de Pareto verdadero de ZDT1

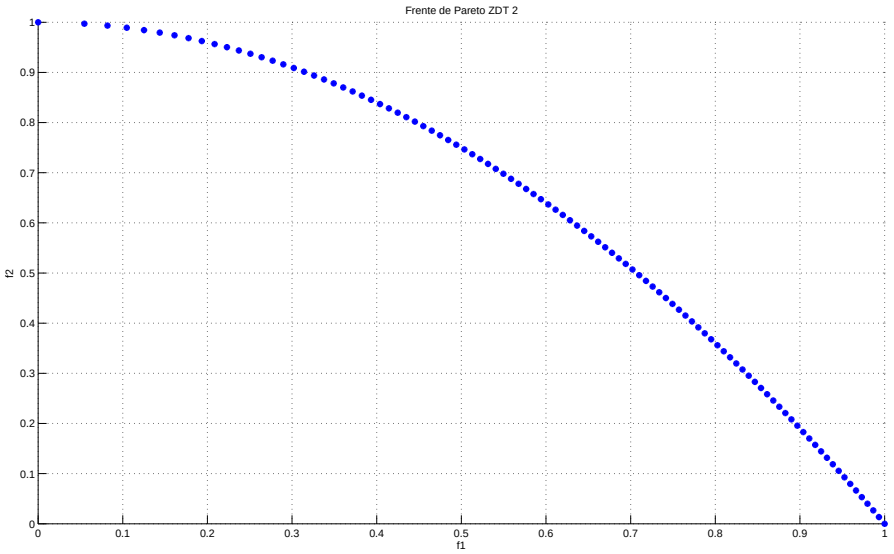

Figura 3.2: Frente de Pareto verdadero de ZDT2

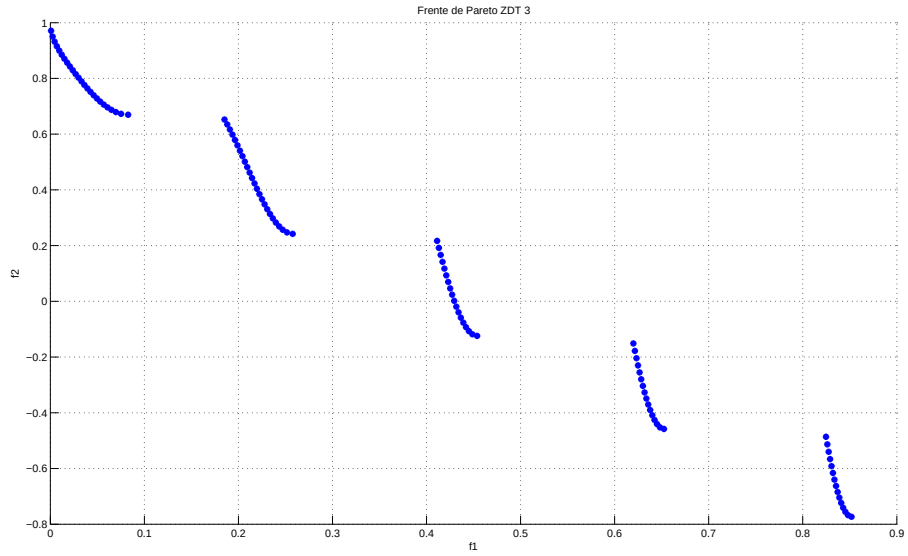

Figura 3.3: Frente de Pareto verdadero de ZDT3

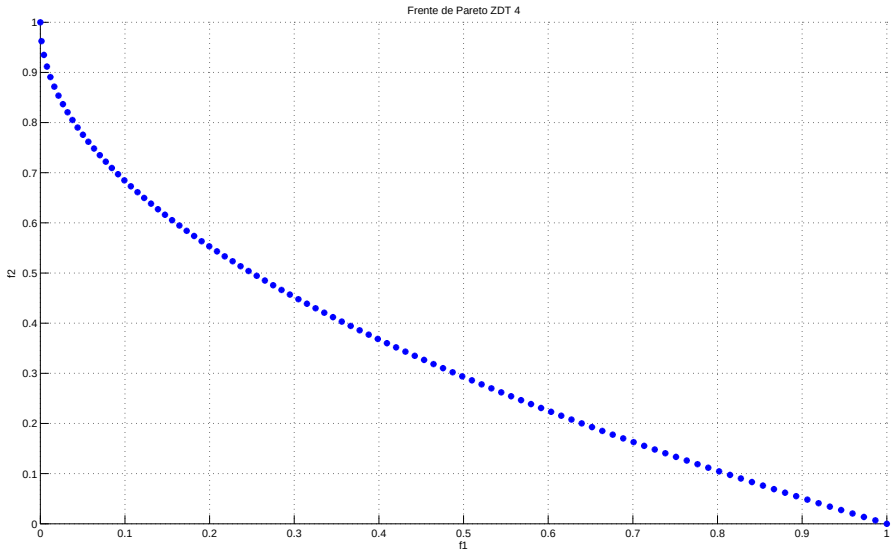

Figura 3.4: Frente de Pareto verdadero de ZDT4

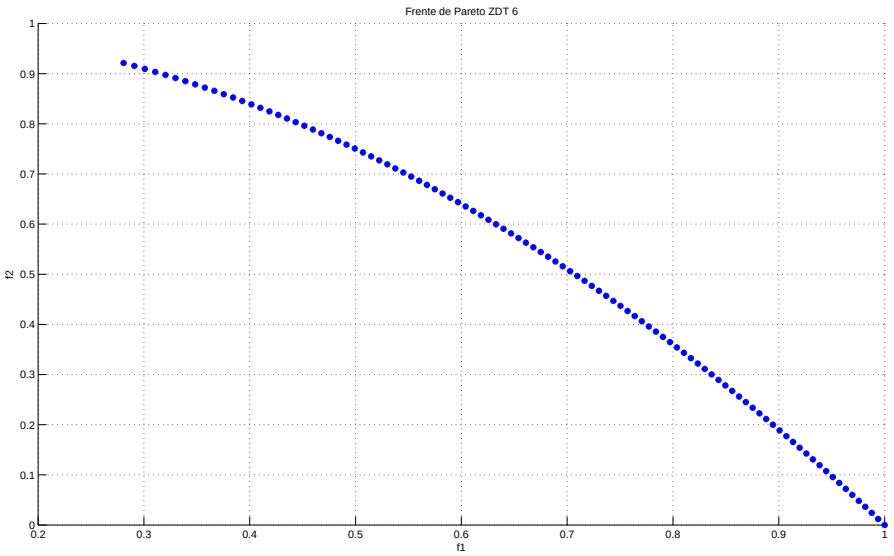

Figura 3.5: Frente de Pareto verdadero de ZDT6

3.2. Evaluación del conjunto de problemas ZDT

La comparación de los resultados de cada algoritmo se realizó con los valores promedio obtenidos por cada métrica. El punto de referencia utilizado en cada caso para el indicador del hipervolumen se muestra en la tabla 3.4.

Los resultados obtenidos para los primeros tres problemas ZDT son similares para a mayoria de los resultados. Donde el SMS-EMOA resulta ganador en la mayor parte de las métricas convergencia seguido por nuestra propuesta (MOPSOhv), como se muestra en las tablas 3.5, 3.6 y 3.7. Sin embargo, nuestra propuesta dominada a las soluciones del NSGA-II y MOPSOcd excepto al SMS-EMOA, de acuerdo a la métrica de cobertura.

Conforme a estos resultados se tiene una jerarquía en orden descendente en términos de convergencia en estos primeros tres problemas:

1. SMS-EMOA
2. MOPSOhv
3. NSGA-II
4. MOPSOcd

En estos problemas puede verse que, se tienen dos casos especiales (ZDT4 y ZDT6).

El problema ZDT4 tiene un frente de Pareto que es altamente multifrontal el cual consiste en 21^9 frentes de Paretos locales. Nuestra propuesta (MOPSOhv) no es capaz de generar buenas soluciones del frente quedando rezagado ligeramente por el SMS-EMOA y el NSGA-II (como muestra la tabla 3.8). Sin embargo, obtiene, en general, mejores resultados en la métrica de cobertura las soluciones obtenidas por el SMS-EMOA y marginalmente peores que los del NSGA-II. Los resultados grfícas obtenidas por el algoritmo MOPSOcd

Problema	Punto de referencia
ZDT1	$(1.1, 1.1)$
ZDT2	$(1.1, 1.1)$
ZDT3	$(1.1, 1.1)$
ZDT4	$(0.9, 1.1)$
ZDT6	$(1.1, 1.1)$

Tabla 3.4: Puntos de referencia utilizados para el conjunto de problemas ZDT

muestran que queda atrapado en un frente de Pareto local, por lo que, nuestra propuesta es mejor.

Los resultados gráficos de las figuras 3.1, 3.2 y 3.3 muestran que el SMS-EMOA se distribuye mejor en la mayoria de esto problemas seguido por nuestra propuesta (MOPSOhv). En la figura 3.4 muestra como el MOPSOcd queda atrapado en un frete de Pareto local.

El problema ZDT6 presenta un espacio de búsqueda no uniforme. Los resultados obtenidos por nuestra propuesta (MOPSOhv) son ligeramente superados por los obtenidos por el SMS-EMOA y el NSGA-II. Debido a la baja densidad de las soluciones obtenidas por el algoritmo MOPSOcd como se muestra en la figura 3.5 no es capaz de alcanzar una buena aproximación al frente de Pareto real, por lo que, nuestra propuesta es mejor.

Conforme a estos resultados se tiene una jerarquía en orden descendente en términos de convergencia en estos últimos dos problemas:

1. NSGA-II
2. SMS-EMOA
3. MOPSOhv
4. MOPSOcd

En general, las soluciones obtenidas por nuestra propuesta (MOPSOhv), para el conjunto de problemas ZDT, quedan entre las soluciones obtenidas por SMS-EMOA y el NSGA-II, y siendo mejores que las soluciones obtenidas por el MOPSOcd.

Los resultados se muestran de la siguiente forma: el color negro para el mejor valor, el color azul para el segundo, el color verde para el tercero y el color rojo para la peor valor de las métricas.

	Espaciado			
Algoritmo	**Menor**	**Mayor**	**Promedio**	**Desviación**
MOPSOhv	0.003157	0.004218	**0.003524**	**0.000305**
MOPSOcd	0.006072	0.007475	**0.006962**	**0.000362**
NSGA-II	0.005950	0.007813	**0.007020**	**0.000447**
SMS-EMOA	0.001840	0.002976	**0.002320**	**0.000270**
	DGI			
MOPSOhv	0.017011	0.017021	**0.017014**	**0.000003**
MOPSOcd	0.017092	0.017240	**0.017173**	**0.000036**
NSGA-II	0.017008	0.017021	**0.017011**	**0.000003**
SMS-EMOA	0.017008	0.017010	**0.017009**	**0.000001**
	Hipervolumen			
MOPSOhv	0.871620	0.871991	**0.871849**	**0.000095**
MOPSOcd	0.861718	0.868142	**0.864645**	**0.001562**
NSGA-II	0.870107	0.871006	**0.870461**	**0.000206**
SMS-EMOA	0.872115	0.872134	**0.872129**	0.000006
	Cobertura			
Algoritmo	**MOPSOhv**	**MOPSOcd**	**NSGA-II**	**SMS-EMOA**
MOPSOhv	—	**0.722500**	**0.074500**	**0.000001**
MOPSOcd	0.000001	—	0.005500	0.000001
NSGA-II	0.022500	0.618000	—	0.013500
SMS-EMOA	**0.028500**	0.762500	0.053500	—

Tabla 3.5: Resultados correspondientes al problema ZDT1.

La convergencia y la distribución de las soluciones de nuestra propuesta (MOPSOhv) es buena para este problema, seguido por el SMS-EMOA que presenta la mejor convergencia y distribución (como se muestra en la tabla 3.5). Conforme a estos resultados se puede crear una jerarquía en orden descendente en términos de convergencia del conjunto de soluciones próximas al frente de Pareto real:

1. SMS-EMOA
2. MOPSOhv
3. NSGA-II
4. MOPSOcd

También, se observa que en las soluciones de nuestra propuesta dominan a las soluciones del NSGA-II y MOPSOcd, y siendo superado por las soluciones del SMS-EMOA.

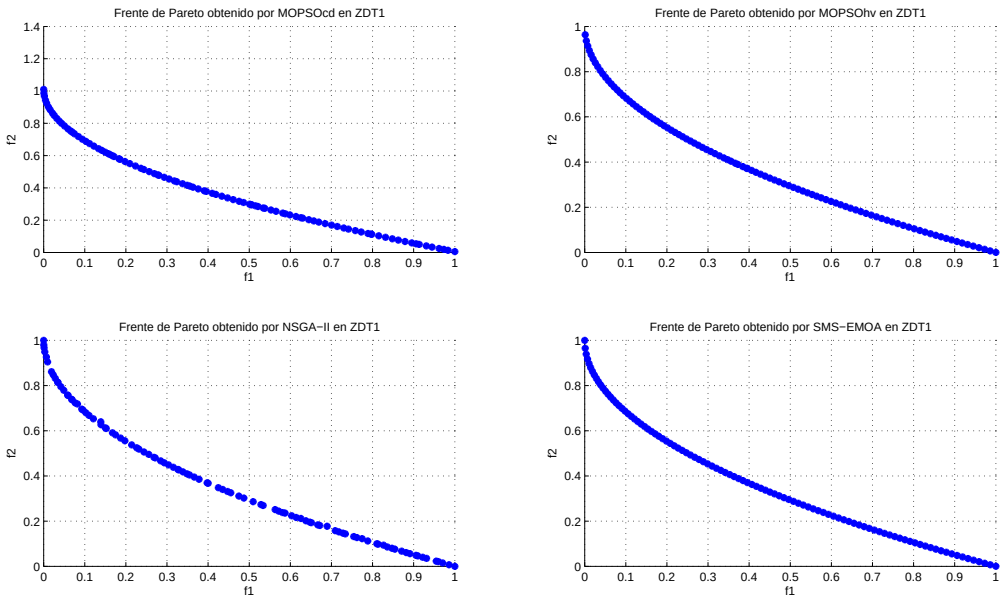

Figura 3.6: Resultados gráficos correspondientes al problema ZDT1.

	Espaciado			
Algoritmo	**Menor**	**Mayor**	**Promedio**	**Desviación**
MOPSOhv	0.004345	0.005129	**0.004679**	**0.000217**
MOPSOcd	0.006054	0.007826	**0.007014**	**0.0005170**
NSGA-II	0.006027	0.008074	**0.006956**	**0.000610**
SMS-EMOA	0.002167	0.004664	**0.004063**	**0.000543**
	DGI			
MOPSOhv	0.027389	0.027394	**0.027391**	**0.000001**
MOPSOcd	0.027462	0.027521	**0.027497**	**0.000015**
NSGA-II	0.027386	0.027389	**0.027387**	**0.000001**
SMS-EMOA	0.027386	0.027387	**0.027386**	**0.000000**
	Hipervolumen			
MOPSOhv	0.538426	0.538666	**0.538568**	**0.000058**
MOPSOcd	0.530732	0.534137	**0.532098**	**0.000882**
NSGA-II	0.537163	0.537699	**0.537424**	**0.000148**
SMS-EMOA	0.538808	0.538879	**0.538872**	**0.000015**
	Cobertura			
Algoritmo	**MOPSOhv**	**MOPSOcd**	**NSGA-II**	**SMS-EMOA**
MOPSOhv	—	**0.637500**	**0.044000**	**0.000500**
MOPSOcd	0.000001	—	0.003500	0.000500
NSGA-II	0.036500	0.589500	—	0.017500
SMS-EMOA	**0.026500**	0.665000	0.034000	—

Tabla 3.6: Resultados correspondientes al problema ZDT2.

La convergencia y la distribución de las soluciones de nuestra propuesta (MOPSOhv) es buena para este problema, seguido por el SMS-EMOA que presenta la mejor convergencia y distribución (como se muestra en la tabla 3.6). Conforme a estos resultados se puede crear una jerarquía en orden descendente en términos de convergencia del conjunto de soluciones próximas al frente de Pareto real:

1. SMS-EMOA
2. MOPSOhv
3. NSGA-II
4. MOPSOcd

También, se observa que en las soluciones de nuestra propuesta dominan a las soluciones del NSGA-II y MOPSOcd, y siendo superado por las soluciones del SMS-EMOA.

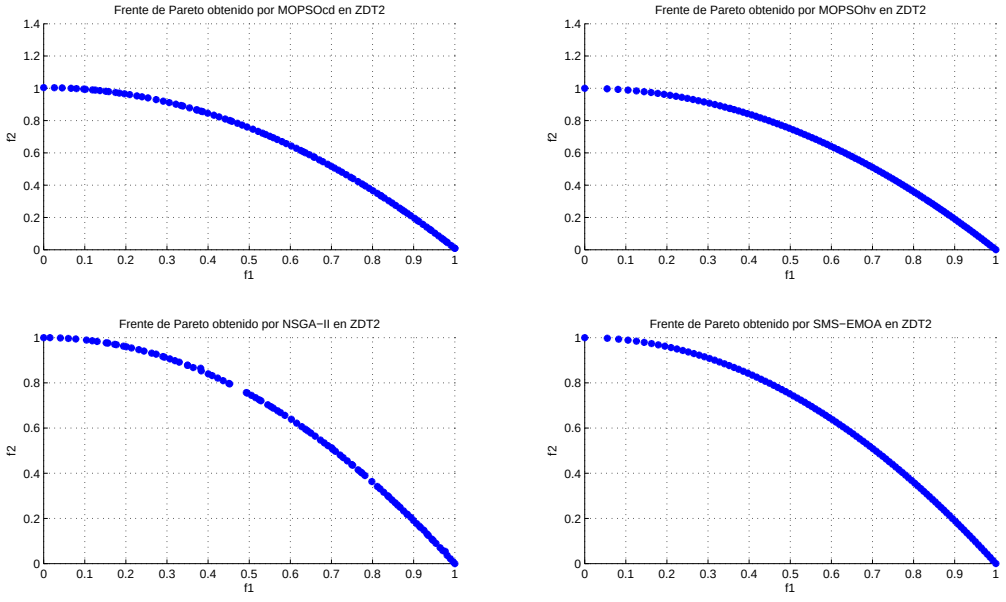

Figura 3.7: Resultados gráficos correspondientes al problema ZDT2.

Algoritmo	Menor	Mayor	Promedio	Desviación
	Espaciado			
MOPSOhv	0.005672	0.006958	**0.006248**	**0.000328**
MOPSOcd	0.007001	0.008605	**0.007656**	**0.000425**
NSGA-II	0.006809	0.009101	**0.007875**	**0.000565**
SMS-EMOA	0.001994	0.002563	**0.002209**	**0.000165**
	DGI			
MOPSOhv	0.011174	0.011219	**0.011198**	**0.000012**
MOPSOcd	0.011389	0.011604	**0.011492**	**0.000064**
NSGA-II	0.011140	0.011184	**0.011166**	**0.000016**
SMS-EMOA	0.019892	0.019894	**0.019892**	**0.000001**
	Hipervolumen			
MOPSOhv	0.952055	0.953776	**0.952907**	**0.000452**
MOPSOcd	0.936835	0.946304	**0.941991**	**0.002510**
NSGA-II	0.953544	0.954219	**0.953912**	**0.000145**
SMS-EMOA	0.522460	0.522466	**0.522464**	**0.000001**
	Cobertura			
Algoritmo	**MOPSOhv**	**MOPSOcd**	**NSGA-II**	**SMS-EMOA**
MOPSOhv	—	**0.637500**	**0.044000**	**0.000500**
MOPSOcd	0.000001	—	0.003500	0.000500
NSGA-II	0.036500	0.589500	—	0.017500
SMS-EMOA	**0.026500**	0.665000	0.034000	—

Tabla 3.7: Resultados correspondientes al problema ZDT3.

La convergencia y la distribución de las soluciones de nuestra propuesta (MOPSOhv) es buena para este problema, seguido por el NSGA-II que presenta la mejor convergencia y el SMS-EMOA presenta la mejor distribución (como se muestra en la tabla 3.7). Conforme a estos resultados se puede crear una jerarquía en orden descendente en términos de convergencia del conjunto de soluciones próximas al frente de Pareto real:

1. NSGA-II
2. MOPSOhv
3. MOPSOcd
4. SMS-EMOA

También, se observa que en las soluciones de nuestra propuesta dominan a las soluciones del NSGA-II y MOPSOcd, y siendo superado por las soluciones del SMS-EMOA.

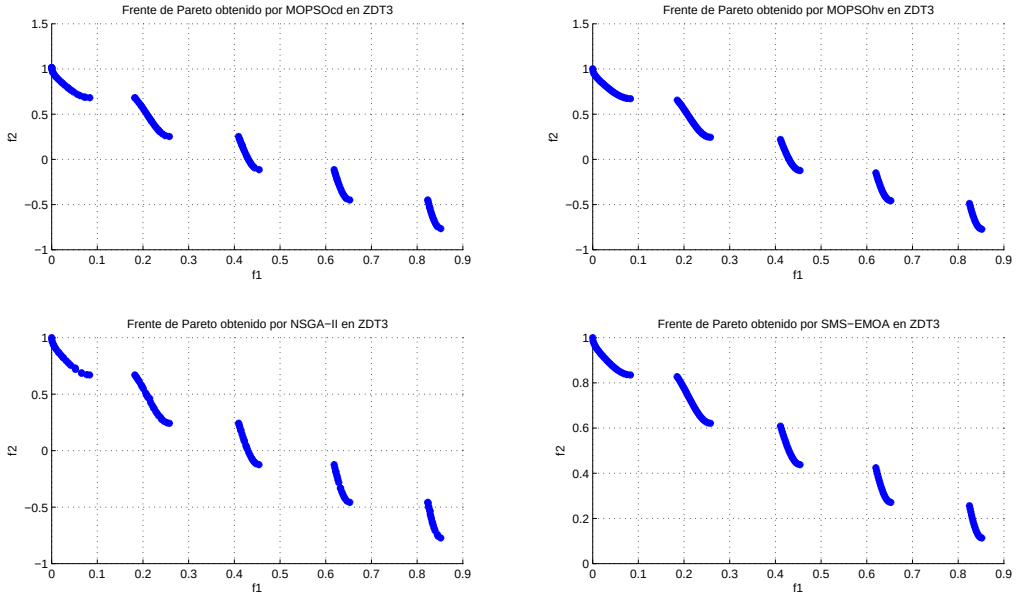

Figura 3.8: Resultados gráficos correspondientes al problema ZDT3.

	Espaciado			
Algoritmo	**Menor**	**Mayor**	**Promedio**	**Desviación**
MOPSOhv	0.004345	0.005129	**0.004679**	**0.000217**
MOPSOcd	0.006054	0.007826	**0.007014**	**0.000517**
NSGA-II	0.007089	0.009201	**0.008052**	**0.000616**
SMS-EMOA	0.002105	0.002981	**0.002492**	**0.000240**
	DGI			
MOPSOhv	0.027389	0.027394	**0.027391**	**0.000001**
MOPSOcd	0.027462	0.027521	**0.027497**	**0.000015**
NSGA-II	0.017008	0.017016	**0.017011**	**0.000002**
SMS-EMOA	0.017008	0.017030	**0.017011**	**0.000004**
	Hipervolumen			
MOPSOhv	0.328865	0.329046	**0.328971**	**0.000052**
MOPSOcd	0.323038	0.325624	**0.324090**	**0.000668**
NSGA-II	0.653158	0.654108	**0.653614**	**0.000237**
SMS-EMOA	0.654266	0.655054	**0.654940**	**0.000165**
	Cobertura			
Algoritmo	**MOPSOhv**	**MOPSOcd**	**NSGA-II**	**SMS-EMOA**
MOPSOhv	—	**1.000000**	**0.022500**	**0.033000**
MOPSOcd	0.000500	—	0.000001	0.000001
NSGA-II	0.004000	1.000000	—	0.024500
SMS-EMOA	0.000001	0.991000	0.001000	—

Tabla 3.8: Resultados del problema ZDT4.

La multimodalidad de ZDT4 causa dificultad a nuestra propuesta (MOPSOhv) para converger de manera óptima, como se muestra en la tabla 3.8. Sin embargo, presenta una buena distribución de las soluciones quedando después del SMS-EMOA. Nuestra propuesta es superada por el SMS-EMOA y NSGA-II y conforme a estos resultados se puede crear una jerarquía en orden descendente en términos de convergencia del conjunto de soluciones próximas al frente de Pareto real:

1. NSGA-II
2. SMS-EMOA
3. MOPSOhv
4. MOPSOcd

También, se observa que en las soluciones de nuestra propuesta dominan a las soluciones de los otros algoritmos.

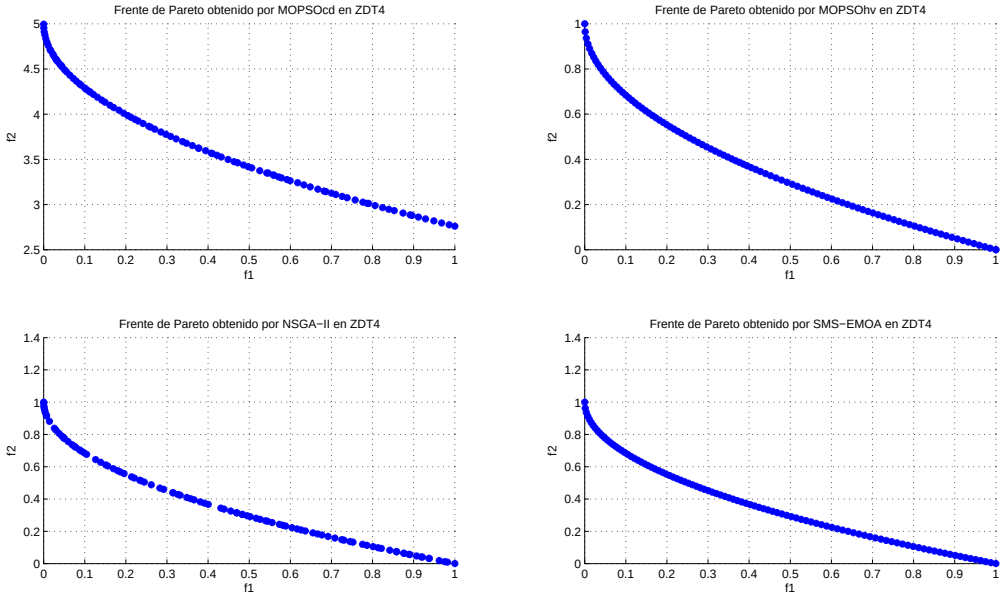

Figura 3.9: Resultados gráficos correspondientes al problema ZDT4.

	Espaciado			
Algoritmo	**Menor**	**Mayor**	**Promedio**	**Desviación**
MOPSOhv	0.001587	0.002728	**0.001975**	**0.000324**
MOPSOcd	0.042774	0.294482	**0.115691**	**0.053160**
NSGA-II	0.005925	0.007914	**0.007055**	**0.000437**
SMS-EMOA	0.000256	0.001011	**0.000603**	**0.000181**
	DGI			
MOPSOhv	0.027380	0.027549	**0.027434**	**0.000046**
MOPSOcd	0.000281	0.000281	**0.000281**	**0.000000**
NSGA-II	0.017011	0.027398	**0.026356**	**0.003115**
SMS-EMOA	0.027390	0.027397	**0.027395**	**0.000002**
	Hipervolumen			
MOPSOhv	0.496939	0.54493	**0.512122**	**0.002083**
MOPSOcd	0.156527	0.479775	**0.383107**	**0.076527**
NSGA-II	0.480422	0.870700	**0.519667**	**0.117011**
SMS-EMOA	0.503836	0.504173	**0.503956**	**0.000093**
	Cobertura			
Algoritmo	**MOPSOhv**	**MOPSOcd**	**NSGA-II**	**SMS-EMOA**
MOPSOhv	—	**0.127778**	**0.000001**	**0.000001**
MOPSOcd	0.059000	—	0.025500	0.019000
NSGA-II	**0.037500**	0.645500	—	0.011000
SMS-EMOA	**0.138889**	0.850000	0.009000	—

Tabla 3.9: Resultados correspondientes al problema ZDT6.

El espacio de los objetivos para ZDT6 muestra una densidad no uniforme causa dificultad a nuestra propuesta (MOPSOhv) para converger de manera óptima, como se muestra en la tabla 3.9. Sin embargo, presenta una buena distribución de las soluciones quedando después del SMS-EMOA y superando al MOPSOcd, ya que este no es capaz de alcanzar una buena aproximación al frente de Pareto real. Nuestra propuesta es superada por el SMS-EMOA y NSGA-II y conforme a estos resultados se puede crear una jerarquía en orden descendente en términos de convergencia del conjunto de soluciones próximas al frente de Pareto real:

1. NSGA-II
2. SMS-EMOA
3. MOPSOhv
4. MOPSOcd

También, se observa que en las soluciones de nuestra propuesta dominan a las soluciones del MOPSOcd y siendo superado por las soluciones de manera ligera por el NSGA-II y SMS-EMOA.

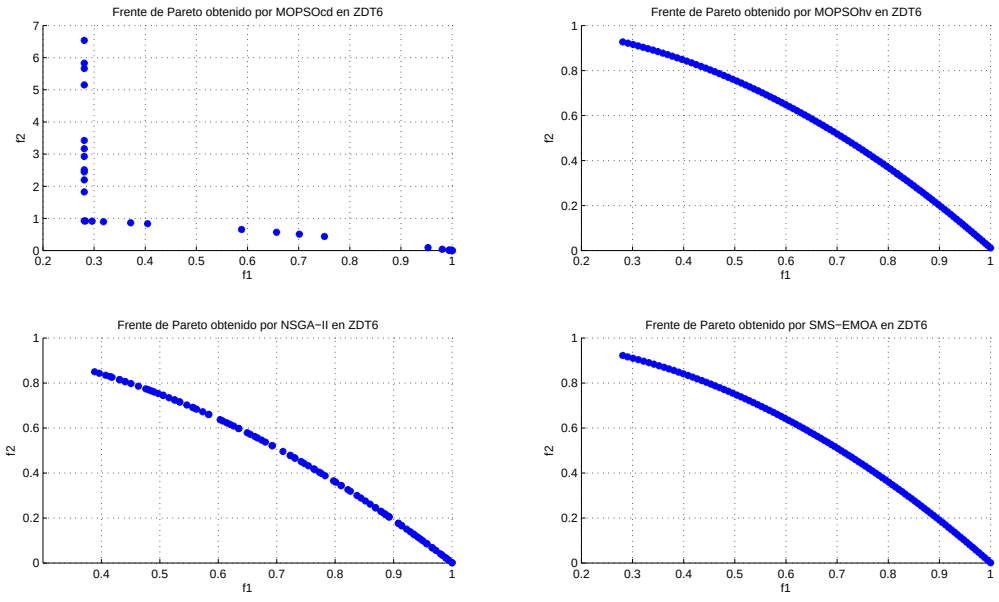

Figura 3.10: Resultados gráficos correspondientes al problema ZDT6.

3.3. Conjunto de problemas DTLZ

Los problemas siguientes son parte de la serie de problemas DTLZ con tres funciones objetivo, tal y como se describen a continuación [Deb et al., 2002]:

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

$$
\begin{aligned}
f_1(x) &= \frac{1}{2} \cdot x_1 \cdot x_2 \cdot \ldots \cdot x_{M-1} \cdot (1 + g(x)) \\
f_2(x) &= \frac{1}{2} \cdot x_1 \cdot x_2 \cdot \ldots \cdot (1 - x_{M-1}) \cdot (1 + g(x)) \\
&\vdots \\
f_M(x) &= \frac{1}{2} \cdot (1 - x_1) \cdot (1 + g(x)) \\
g(x) &= 100 \cdot [k + \sum_{i=M}^{n} (x_i - 0.5)^2 - cos(20 \cdot \pi \cdot (x_i - 0.5))]
\end{aligned}
$$

  donde $n = M + k - 1$ (se sugiere una $k = 5$) y $x_i \in [0, 1]$ con $i = 1, \ldots, n$ (figura 3.11).

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

$$
\begin{aligned}
f_1(x) &= \cos(x_1 \frac{\pi}{2}) \cdot \cos(x_2 \frac{\pi}{2}) \cdot \ldots \cdot \cos(x_{M-1} \frac{\pi}{2}) \cdot (1 + g(x)) \\
f_2(x) &= \cos(x_1 \frac{\pi}{2}) \cdot \cos(x_2 \frac{\pi}{2}) \cdot \ldots \cdot \text{sen}(x_{M-1} \frac{\pi}{2}) \cdot (1 + g(x)) \\
f_3(x) &= \cos(x_1 \frac{\pi}{2}) \cdot \cos(x_2 \frac{\pi}{2}) \cdot \ldots \cdot \text{sen}(x_{M-2} \frac{\pi}{2}) \cdot (1 + g(x)) \\
&\vdots \\
f_{M-1}(x) &= \cos(x_1 \frac{\pi}{2}) \cdot \text{sen}(x_2 \frac{\pi}{2})(1 + g(x)) \\
f_M(x) &= \text{sen}(x_1 \frac{\pi}{2}) \cdot (1 + g(x)) \\
g(x) &= \sum_i = (x_i - 0.5)^2
\end{aligned}
$$

  donde $n = M + k - 1$ (se sugiere una $k = 10$) y $x_i \in [0, 1]$ con $i = 1, \ldots, n$. Este problema se puede utilizar para investigar la escalabilidad (figura 3.12).

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

$$
\begin{aligned}
f_1(x) &= \cos(x_1\frac{\pi}{2}) \cdot \cos(x_2\frac{\pi}{2}) \cdot \ldots \cdot \cos(x_{M-1}\frac{\pi}{2}) \cdot (1+g(x)) \\
f_2(x) &= \cos(x_1\frac{\pi}{2}) \cdot \cos(x_2\frac{\pi}{2}) \cdot \ldots \cdot \operatorname{sen}(x_{M-1}\frac{\pi}{2}) \cdot (1+g(x)) \\
f_3(x) &= \cos(x_1\frac{\pi}{2}) \cdot \cos(x_2\frac{\pi}{2}) \cdot \ldots \cdot \operatorname{sen}(x_{M-2}\frac{\pi}{2}) \cdot (1+g(x)) \\
&\vdots \\
f_{M-1}(x) &= \cos(x_1\frac{\pi}{2}) \cdot \operatorname{sen}(x_2\frac{\pi}{2}) \cdot (1+g(x)) \\
f_M(x) &= \operatorname{sen}(x_1\frac{\pi}{2}) \cdot (1+g(x)) \\
g(x) &= 100 \cdot [k + \sum_{i=M}^{n} (x_i - 0.5)^2 - \cos(20 \cdot \pi \cdot (x_i - 0.5))]
\end{aligned}
$$

  donde $n = M + k - 1$ (se sugiere una $k = 10$) y $x_i \in [0, 1]$ con $i = 1, \ldots, n$. La forma del frente de Pareto de este problema es similar al del problema DTLZ2 (figura 3.12).

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

$$
\begin{aligned}
f_1(x) &= \cos(x_1^\alpha\frac{\pi}{2}) \cdot \cos(x_2^\alpha\frac{\pi}{2}) \cdot \ldots \cdot \cos(x_{M-1}^\alpha\frac{\pi}{2}) \cdot (1+g(x)) \\
f_2(x) &= \cos(x_1^\alpha\frac{\pi}{2}) \cdot \cos(x_2^\alpha\frac{\pi}{2}) \cdot \ldots \cdot \operatorname{sen}(x_{M-1}^\alpha\frac{\pi}{2}) \cdot (1+g(x)) \\
f_3(x) &= \cos(x_1^\alpha\frac{\pi}{2}) \cdot \cos(x_2^\alpha\frac{\pi}{2}) \cdot \ldots \cdot \operatorname{sen}(x_{M-2}^\alpha\frac{\pi}{2}) \cdot (1+g(x)) \\
&\vdots \\
f_{M-1}(x) &= \cos(x_1^\alpha\frac{\pi}{2}) \cdot \operatorname{sen}(x_2^\alpha\frac{\pi}{2}) \cdot (1+g(x)) \\
f_M(x) &= \operatorname{sen}(x_1^\alpha\frac{\pi}{2}) \cdot (1+g(x)) \\
g(x) &= \sum_i (x_i - 0.5)^2
\end{aligned}
$$

  donde $n = M + k - 1$ (se sugiere una $k = 10$ y $\alpha = 100$) y $x_i \in [0, 1]$ con $i = 1, \ldots, n$. Este problema prueba la habilidad de mantener una buena distribución de las soluciones. La forma del frente de Pareto de este problema es similar al del problema DTLZ2 (figura 3.12).

- **L**a función de prueba **DTLZ5** tiene un frente de Pareto curvo:

$$
\begin{aligned}
f_1(x) &= \cos(\theta_1\frac{\pi}{2}) \cdot \cos(\theta_2\frac{\pi}{2}) \cdot \ldots \cdot \cos(\theta_{M-1}\frac{\pi}{2}) \cdot (1+g(x)) \\
f_2(x) &= \cos(\theta_1\frac{\pi}{2}) \cdot \cos(\theta_2\frac{\pi}{2}) \cdot \ldots \cdot \operatorname{sen}(\theta_{M-1}\frac{\pi}{2}) \cdot (1+g(x)) \\
f_3(x) &= \cos(\theta_1\frac{\pi}{2}) \cdot \cos(\theta_2\frac{\pi}{2}) \cdot \ldots \cdot \operatorname{sen}(\theta_{M-2}\frac{\pi}{2}) \cdot (1+g(x)) \\
&\vdots \\
f_{M-1}(x) &= \cos(\theta_1\frac{\pi}{2}) \cdot \operatorname{sen}(\theta_2\frac{\pi}{2}) \cdot (1+g(x)) \\
f_M(x) &= \operatorname{sen}(\theta_1\frac{\pi}{2}) \cdot (1+g(x)) \\
\theta_1 &= \frac{\pi}{2}x_1 \\
\theta_i &= \frac{\pi}{4\cdot(1+g(x))}(1+2\cdot g(x)\cdot x_i), \text{para } i = 2,3\ldots,(M-1) \\
g(x) &= \sum_i^n (x_i - 0.5)^2
\end{aligned}
$$

donde $n = M + k - 1$ (se sugiere una $k = 10$) y $x_i \in [0,1]$ con $i = 1,\ldots,n$ (figura 3.13).

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

$$
\begin{aligned}
f_1(x) &= \cos(\theta_1\frac{\pi}{2}) \cdot \cos(\theta_2\frac{\pi}{2}) \cdot \ldots \cdot \cos(\theta_{M-1}\frac{\pi}{2}) \cdot (1+g(x)) \\
f_2(x) &= \cos(\theta_1\frac{\pi}{2}) \cdot \cos(\theta_2\frac{\pi}{2}) \cdot \ldots \cdot \operatorname{sen}(\theta_{M-1}\frac{\pi}{2}) \cdot (1+g(x)) \\
f_3(x) &= \cos(\theta_1\frac{\pi}{2}) \cdot \cos(\theta_2\frac{\pi}{2}) \cdot \ldots \cdot \operatorname{sen}(\theta_{M-2}\frac{\pi}{2}) \cdot (1+g(x)) \\
&\vdots \\
f_{M-1}(x) &= \cos(\theta_1\frac{\pi}{2}) \cdot \operatorname{sen}(\theta_2\frac{\pi}{2}) \cdot (1+g(x)) \\
f_M(x) &= \operatorname{sen}(\theta_1\frac{\pi}{2}) \cdot (1+g(x)) \\
\theta_1 &= \frac{\pi}{2}x_1 \\
\theta_i &= \frac{\pi}{4(1+g(x))}(1+2\cdot g(x)\cdot x_i), \text{para } i = 2,3\ldots,(M-1) \\
g(x) &= \sum_i^n = (x_i - 0.5)^0.1
\end{aligned}
$$

donde $n = M + k - 1$ (se sugiere una $k = 10$) y $x_i \in [0,1]$ con $i = 1,\ldots,n$ (figura 3.14).

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

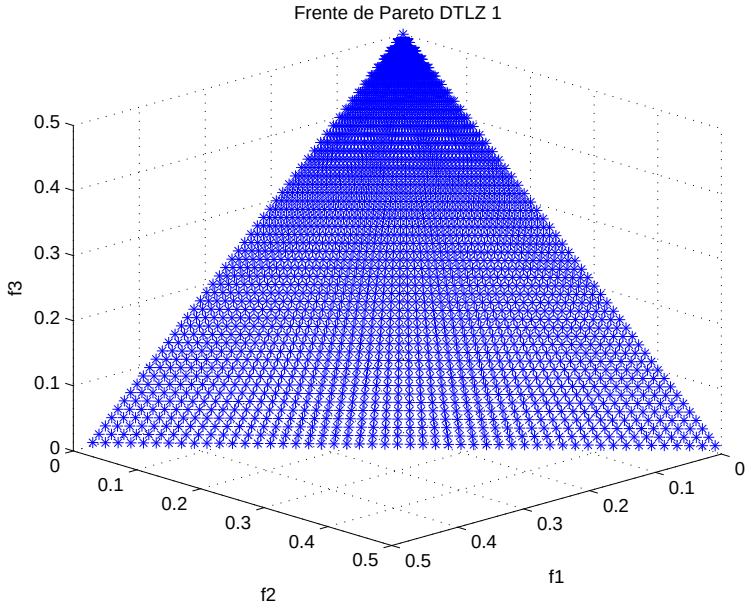

Figura 3.11: Frente de Pareto verdadero de DTLZ1

$$
\begin{aligned}
f_1(x) &= x_1 \\
f_2(x) &= x_2 \\
&\vdots \\
f_{M-1}(x) &= x_{M-1} \\
f_M(x) &= (1 + g(x_M)) \cdot h(f_1, f_2, \ldots, f_{M-1} g(x)) \\
g(x) &= 1 + \frac{9}{k} \cdot \sum_{i=2}^{n} x_i \\
h(f_1, f_2, \ldots, f_{M-1} g(x)) &= M - \sum_{i=1}^{M-1} \left(\frac{f_i}{1 + g(x)} (1 + \operatorname{sen}(3 \cdot \pi \cdot f_i))\right)
\end{aligned}
$$

donde $n = M + k - 1$ (se sugiere una $k = 20$) y $x_i \in [0, 1]$ con $i = 1, \ldots, n$. Este problema prueba la habilidad de mantener individuos en las diferentes regiones del frente de Pareto (figura 3.15).

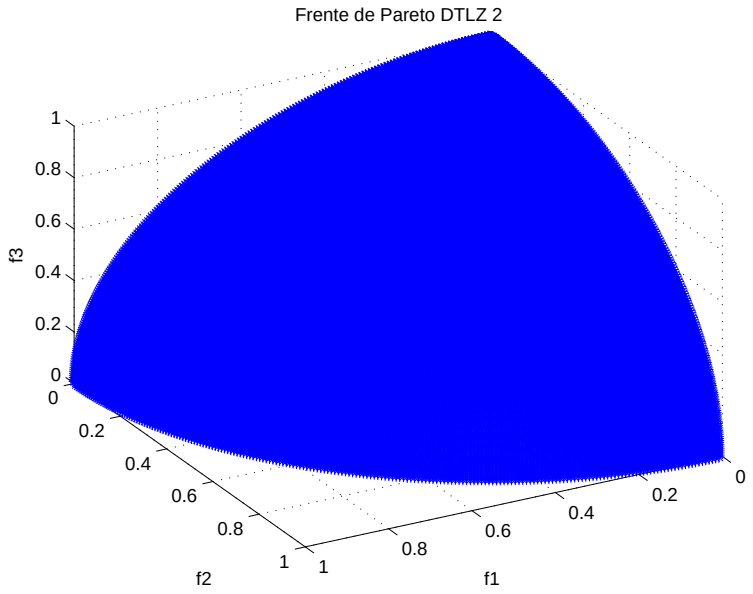

Figura 3.12: Frente de Pareto verdadero de DTLZ2

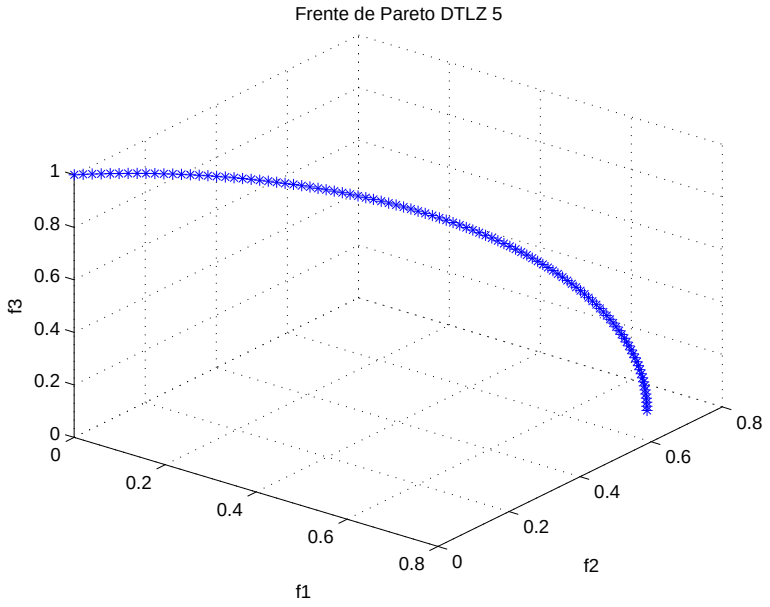

Figura 3.13: Frente de Pareto verdadero de DTLZ5

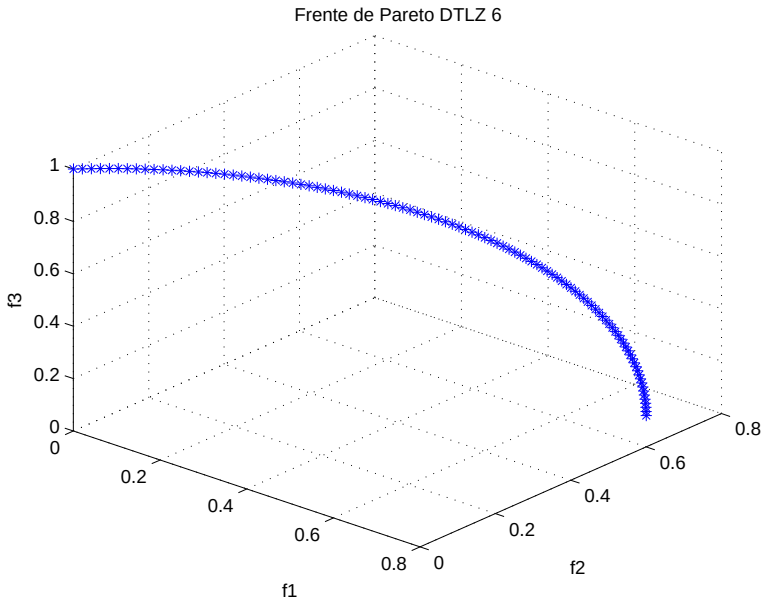

Figura 3.14: Frente de Pareto verdadero de DTLZ6

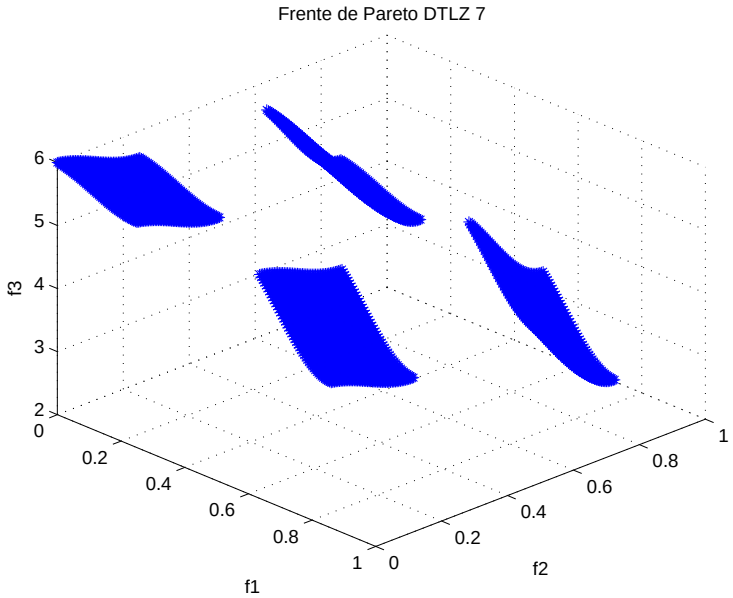

Figura 3.15: Frente de Pareto verdadero de DTLZ7

3.4. Evaluación del conjunto de problemas DTLZ

Para calcular la métrica del hipervolumen se utiliza como punto de referencia el mostrado en la tabla 3.10.

Los resultados obtenidos por nuestra propuesta (MOPSOhv) para los problemas DTLZ1, DTLZ3 y DTLZ6 son mejores que los obtenidos por el MOPSOcd, ya que las soluciones de nuestra propuesta, según la métrcia de cobertura, dominan completamente las soluciones arrojadas por MOPSOcd. Sin embargo, los resultados de nuestra propuesta son malos en comparación con los obtenidos por SMS-EMOA y NSGA-II, los cuales obtuvieron soluciones que dominan completamente a las nuestras como se muestra en las tablas 3.11, 3.13 y 3.16.

Conforme a estos resultados se tiene una jerarquía en orden descendente en términos de convergencia en estos primeros tres problemas:

1. NSGA-II
2. SMS-EMOA
3. MOPSOhv
4. MOPSOcd

El problema DTLZ4 presenta un frente de Pareto cóncavo. En la la figura 3.19 se muestran como los resultados obtenidos por nuestra propuesta (MOPSOhv) se distribuyen hacia los extremos del frente de Pareto, lo que causa tener una menor cobertura sobre éste y causando que el valor en la métrica del hipervolumen sea menor que los otros algoritmos. Sin embargo, las soluciones dominan a las soluciones de los demás algoritmos como se muestra en la tabla 3.14.

El problema DTLZ5 presenta un frente de Pareto curvo. DTLZ5 se considera un problema fácil, ya que su espacio de búsqueda presenta sesgo hacia soluciones cercanas al frente de Pareto verdadero. La figura 3.20 muestra que los resultados son similares para todos los algoritmos. Sin embargo, nuestras soluciones (MOPSOhv) dominan a las soluciones de los demás algoritmos excepto por las de SMS-EMOA como se muestra en la tabla 3.15.

El problema DTLZ7 presenta un frente de Pareto con regiones discontinuas. Los resultados de nuestra propuesta (MOPSOhv) son similares que los de los algoritmos NSGA-II y MOPSOcd. Sin embargo, nuestras soluciones son dominadas por las obtenidas por SMS-EMOA (tabla 3.17). La figura 3.22 muestra como las soluciones se concentran en una parte del frente de Pareto.

Conforme a estos resultados se tiene una jerarquía general en orden descendente en términos de convergencia para DTLZ4, DTLZ5 y DTLZ7 problemas:

1. NSGA-II
2. SMS-EMOA
3. MOPSOhv
4. MOPSOcd

Problema	Punto de referencia
DTLZ1	$(1.1, 1.1, 1.1)$
DTLZ2	$(1.1, 1.1, 1.1)$
DTLZ3	$(5.0, 5.0, 5.0)$
DTLZ4	$(1.1, 1.1, 1.1)$
DTLZ5	$(1.1, 1.1, 1.1)$
DTLZ6	$(1.1, 1.1, 1.1)$
DTLZ7	$(1.1, 1.1, 7.0)$

Tabla 3.10: Puntos de referencia utilizados para el conjunto de problemas DTLZ

	Espaciado			
Algoritmo	**Menor**	**Mayor**	**Promedio**	**Desviación**
MOPSOhv	0.026239	0.184637	**0.081790**	**0.045829**
MOPSOcd	0.531066	1.547502	**0.974734**	**0.295577**
NSGA-II	0.015203	0.023959	**0.019360**	**0.002384**
SMS-EMOA	0.005793	0.012914	**0.006947**	**0.001461**
	DGI			
MOPSOhv	0.000617	0.035965	**0.013359**	**0.011381**
MOPSOcd	0.145461	0.462595	**0.269641**	**0.080597**
NSGA-II	0.000540	0.003699	**0.001261**	**0.001148**
SMS-EMOA	0.005777	0.011575	**0.006088**	**0.001259**
	Hipervolumen			
MOPSOhv	0.000000	1.290672	**0.838140**	**0.351522**
MOPSOcd	0.000000	0.000000	**0.000000**	**0.000000**
NSGA-II	1.244198	1.297406	**1.270538**	**0.016839**
SMS-EMOA	1.123310	1.305100	**1.295857**	**0.039586**
	Cobertura			
Algoritmo	**MOPSOhv**	**MOPSOcd**	**NSGA-II**	**SMS-EMOA**
MOPSOhv	—	**0.964500**	**0.000000**	**0.000000**
MOPSOcd	0.000000	—	0.000000	0.000000
NSGA-II	**0.838500**	0.977500	—	0.039000
SMS-EMOA	**0.771000**	0.911500	0.000000	—

Tabla 3.11: Resultados correspondientes al problema DTLZ1.

La multimodalidad de DTLZ1 causa dificultad a nuestra propuesta (MOSPSOhv) para converger de manera óptima, como se muestra en la tabla 3.11. En la gráfica 3.16 se muestra que el MOPSOcd no tiene una buena convergencia al frente de Pareto y no muestra una buena distribución de las soluciones, por lo que nuestra propuesta se mantienen arriba de éste. Sin embargo, nuestra propuesta es superada por el SMS-EMOA y NSGA-II y conforme a estos resultados se puede crear una jerarquía en orden descendente en términos de convergencia del conjunto de soluciones próximas al frente de Pareto real:

1. NSGA-II
2. SMS-EMOA
3. MOPSOhv
4. MOPSOcd

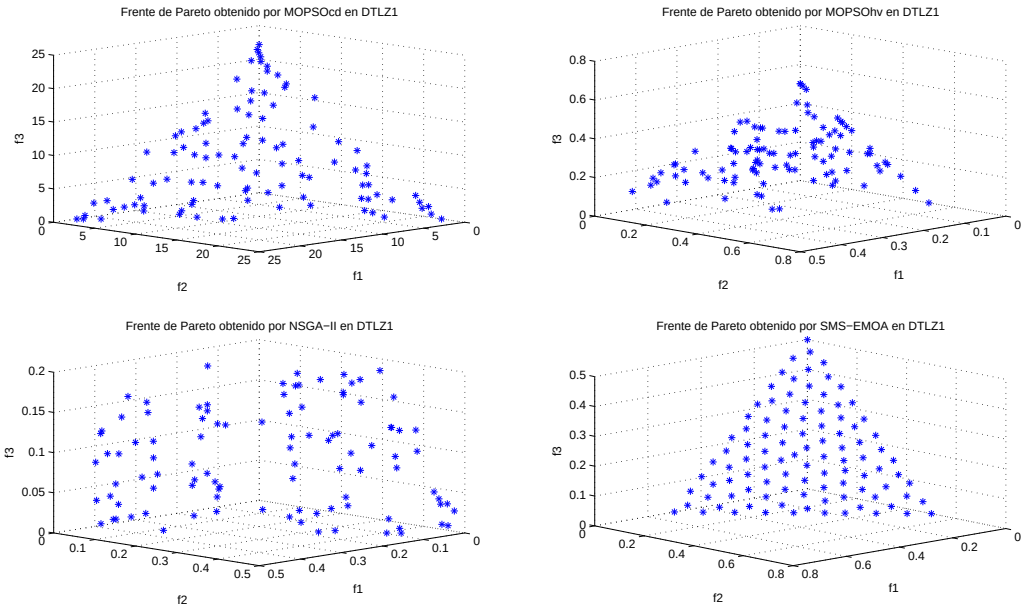

Figura 3.16: Resultados gráficos correspondientes al problema DTLZ1.

	Espaciado			
Algoritmo	**Menor**	**Mayor**	**Promedio**	**Desviación**
MOPSOhv	0.031912	0.087823	**0.056173**	**0.012780**
MOPSOcd	0.042795	0.064965	**0.052077**	**0.005154**
NSGA-II	0.043968	0.065239	**0.055460**	**0.005003**
SMS-EMOA	0.040210	0.047397	**0.042716**	**0.001942**
	DGI			
MOPSOhv	0.000421	0.003117	**0.000806**	**0.000611**
MOPSOcd	0.000353	0.000430	**0.000383**	**0.000022**
NSGA-II	0.000354	0.000437	**0.000380**	**0.000018**
SMS-EMOA	0.005000	0.005000	**0.005000**	**0.000000**
	Hipervolumen			
MOPSOhv	0.477781	0.695914	**0.626972**	**0.046583**
MOPSOcd	0.637661	0.700693	**0.669615**	**0.019100**
NSGA-II	0.676094	0.709199	**0.697212**	**0.007736**
SMS-EMOA	0.757997	0.758163	**0.758091**	**0.000047**
	Cobertura			
Algoritmo	**MOPSOhv**	**MOPSOcd**	**NSGA-II**	**SMS-EMOA**
MOPSOhv	—	**0.225500**	**0.009000**	**0.000000**
MOPSOcd	0.000000	—	0.001500	0.000000
NSGA-II	0.001000	0.296000	—	0.010000
SMS-EMOA	**0.003000**	0.444000	0.006000	—

Tabla 3.12: Resultados correspondientes al problema DTLZ2.

En el problema DTLZ2 se observa que en nuestra propuesta (MOPSOhv) no tiene una distribución de las soluciones, lo que causa que existan espacios grandes en la aproximación al frente de Pareto, como se observa en la figura 3.17, lo que causa que no se tenga una mayor cobertura del frente de Pareto, por lo tanto, el valor de la métrica del hipervolumen disminuya, como se observa en la tabla 3.12. Esto se debe, a que nuestra propuesta utiliza una métrica que aproxima las contribuciones al hipervolumen para reemplazar soluciones en el archivo, por lo que, disminuye la calidad de las soluciones alcanzadas por éste. Nuestra propuesta es superada por el SMS-EMOA y NSGA-II y conforme a estos resultados se puede crear una jerarquía en orden descendente en términos de convergencia del conjunto de soluciones próximas al frente de Pareto real:

1. SMS-EMOA
2. NSGA-II
3. MOPSOhv
4. MOPSOcd

También, se observa que en las soluciones de nuestra propuesta dominan a las soluciones del NSGA-II y MOPSOcd, y siendo superado por las soluciones del SMS-EMOA.

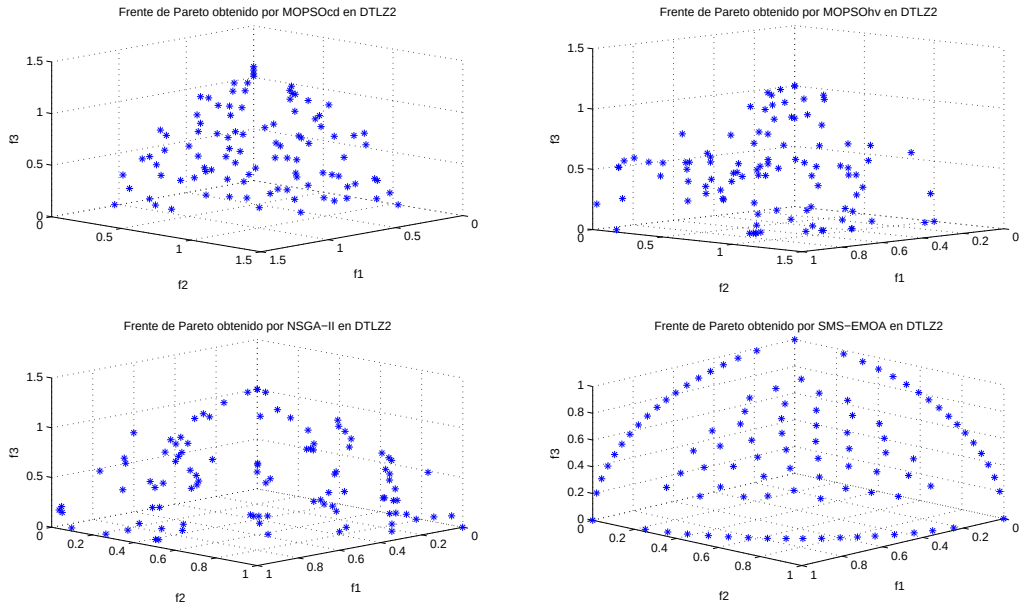

Figura 3.17: Resultados gráficos correspondientes al problema DTLZ2.

	Espaciado			
Algoritmo	**Menor**	**Mayor**	**Promedio**	**Desviación**
MOPSOhv	0.523852	6.539527	**1.759328**	**1.322038**
MOPSOcd	0.774392	5.346362	**2.143857**	**0.915538**
NSGA-II	0.045404	0.067297	**0.056032**	**0.004531**
SMS-EMOA	0.037710	0.083937	**0.043889**	**0.009394**
	DGI			
MOPSOhv	0.187750	1.484958	**0.594644**	**0.302257**
MOPSOcd	0.199480	1.812156	**0.630432**	**0.323953**
NSGA-II	0.001123	0.001441	**0.001235**	**0.000083**
SMS-EMOA	0.015616	0.031270	**0.016425**	**0.003406**
	Hipervolumen			
MOPSOhv	0.000000	0.000000	**0.000000**	**0.000000**
MOPSOcd	0.000000	0.000000	**0.000000**	**0.000000**
NSGA-II	123.004398	124.174476	**123.673815**	**0.295556**
SMS-EMOA	120.390195	124.425951	**124.221079**	**0.878873**
	Cobertura			
Algoritmo	**MOPSOhv**	**MOPSOcd**	**NSGA-II**	**SMS-EMOA**
MOPSOhv	—	**0.546000**	**0.000000**	**0.000000**
MOPSOcd	0.215500	—	0.000000	0.000000
NSGA-II	**1.000000**	0.980000	—	0.052000
SMS-EMOA	**0.960000**	0.940000	0.000000	—

Tabla 3.13: Resultados correspondientes al problema DTLZ3.

La multimodalidad de DTLZ3 causa dificultad a nuestra propuesta (MOPSOhv) para converger de manera óptima, como se muestra en la tabla 3.13. En la gráfica 3.18 se muestra que el MOPSOcd y nuestra propuesta no tiene una buena convergencia al frente de Pareto y no muestran una buena distribución de las soluciones. Sin embargo, nuestra propuesta domina a las soluciones del MOPSOcd. Nuestra propuesta es superada por el SMS-EMOA y NSGA-II y conforme a estos resultados se puede crear una jerarquía en orden descendente en términos de convergencia del conjunto de soluciones próximas al frente de Pareto real:

1. NSGA-II
2. SMS-EMOA
3. MOPSOhv
4. MOPSOcd

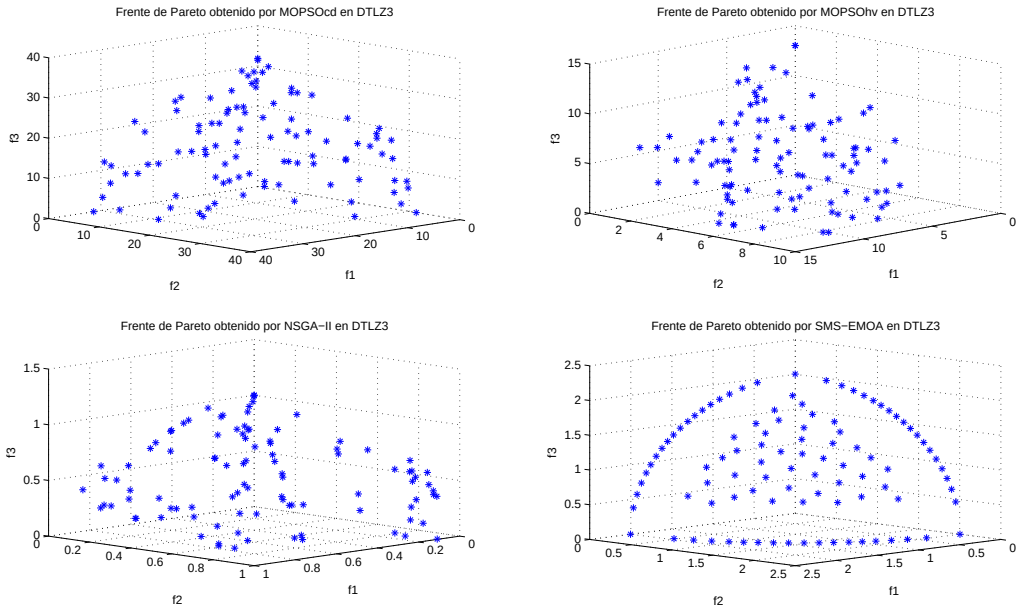

Figura 3.18: Resultados gráficos correspondientes al problema DTLZ3.

	Espaciado			
Algoritmo	**Menor**	**Mayor**	**Promedio**	**Desviación**
MOPSOhv	0.009443	0.080015	**0.047999**	**0.021016**
MOPSOcd	0.052363	0.061539	**0.056542**	**0.002922**
NSGA-II	0.000000	0.063012	**0.051862**	**0.013087**
SMS-EMOA	0.039215	0.045833	**0.042030**	**0.001928**
	DGI			
MOPSOhv	0.003982	0.010549	**0.006694**	**0.002366**
MOPSOcd	0.001142	0.001271	**0.001208**	**0.000037**
NSGA-II	0.001097	0.015257	**0.001875**	**0.003071**
SMS-EMOA	0.015606	0.015606	**0.015606**	**0.000000**
	Hipervolumen			
MOPSOhv	0.337305	0.666145	**0.548703**	**0.117426**
MOPSOcd	0.668445	0.708459	**0.688920**	**0.010033**
NSGA-II	0.121000	0.715370	**0.674632**	**0.127135**
SMS-EMOA	0.757966	0.758160	**0.758069**	**0.000050**
	Cobertura			
Algoritmo	**MOPSOhv**	**MOPSOcd**	**NSGA-II**	**SMS-EMOA**
MOPSOhv	—	**0.189000**	**0.013500**	**0.017000**
MOPSOcd	0.000000	—	0.000500	0.000000
NSGA-II	0.008421	0.199500	—	0.000000
SMS-EMOA	0.000000	0.275500	0.009500	—

Tabla 3.14: Resultados correspondientes al problema DTLZ4.

En el problema DTLZ4 se observa que en las soluciones de nuestra propuesta (MOPSOhv) la distribución de las soluciones se concentren en los extremos, causando que existan espacios muy grandes en el centro del frente de Pareto, como se observa en la figura 3.19, y como consecuencia el valor de la métrica del hipervolumen disminuya, como se observa en la tabla 3.14. Esto puede ser por que nuestra propuesta se concentra en los extremos del frente, ya que estos, pueden ser aquellas partículas que guían la búsqueda de nuevas solciones, según su contribución al hipervolumen. Nuestra propuesta es superada por el SMS-EMOA y NSGA-II y conforme a estos resultados se puede crear una jerarquía en orden descendente en términos de convergencia del conjunto de soluciones próximas al frente de Pareto real:

1. SMS-EMOA
2. NSGA-II
3. MOPSOhv
4. MOPSOcd

También, se observa que en las soluciones de nuestra propuesta dominan a las soluciones de los otros algoritmos.

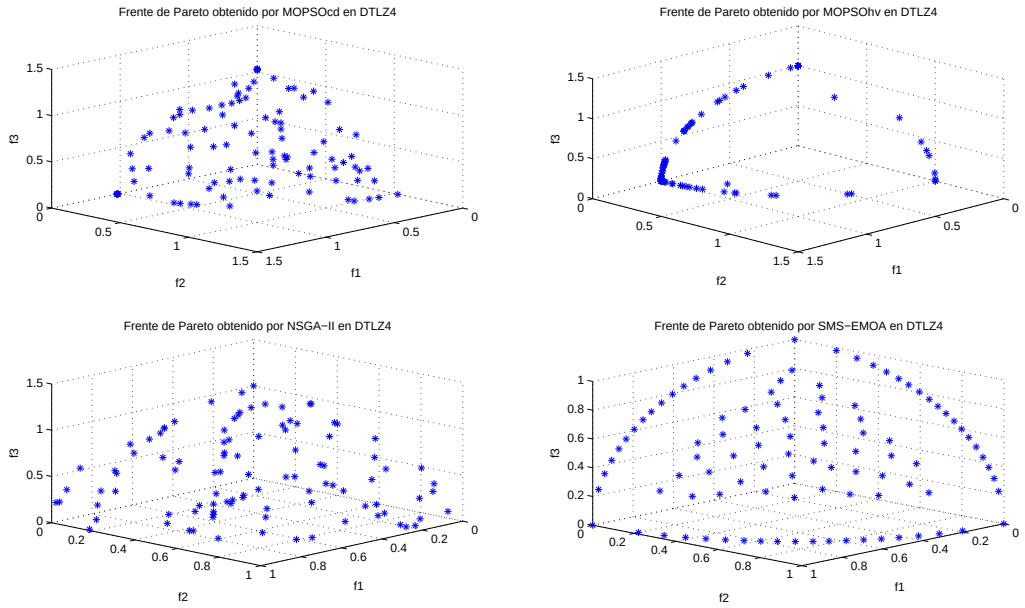

Figura 3.19: Resultados gráficos correspondientes al problema DTLZ4.

	Espaciado			
Algoritmo	**Menor**	**Mayor**	**Promedio**	**Desviación**
MOPSOhv	0.009066	0.027088	**0.016552**	**0.005419**
MOPSOcd	0.007105	0.009411	**0.008436**	**0.000610**
NSGA-II	0.008281	0.011557	**0.009727**	**0.000757**
SMS-EMOA	0.008044	0.009672	**0.008763**	**0.000463**
	DGI			
MOPSOhv	0.000125	0.001589	**0.000439**	**0.000390**
MOPSOcd	0.000049	0.000058	**0.000054**	**0.000002**
NSGA-II	0.000061	0.000083	**0.000069**	**0.000005**
SMS-EMOA	0.009901	0.009901	**0.009901**	**0.000000**
	Hipervolumen			
MOPSOhv	0.396024	0.435369	**0.428426**	**0.010028**
MOPSOcd	0.438762	0.438946	**0.438852**	**0.000056**
NSGA-II	0.437414	0.438274	**0.437990**	**0.000223**
SMS-EMOA	0.439348	0.439390	**0.439374**	**0.000011**
	Cobertura			
Algoritmo	**MOPSOhv**	**MOPSOcd**	**NSGA-II**	**SMS-EMOA**
MOPSOhv	—	**0.024500**	**0.024500**	**0.000000**
MOPSOcd	0.006500	—	0.034500	0.010000
NSGA-II	0.020500	0.000000	—	0.010000
SMS-EMOA	**0.000500**	0.000000	0.022000	—

Tabla 3.15: Resultados correspondientes al problema DTLZ5.

En el problema DTLZ5 se observa que en nuestra propuesta (MOPSOhv) no tiene una distribución de las soluciones, lo que causa que existan espacios en la aproximación al frente de Pareto, como se observa en la figura 3.17, lo que causa que no se tenga una buena cobertura del frente de Pareto, por lo tanto, el valor de la métrica del hipervolumen disminuya, como se observa en la tabla 3.12. Nuestra propuesta es superada por el SMS-EMOA y NSGA-II y conforme a estos resultados se puede crear una jerarquía en orden descendente en términos de convergencia del conjunto de soluciones próximas al frente de Pareto real:

1. NSGA-II
2. SMS-EMOA
3. MOPSOcd
4. MOPSOhv

También, se observa que en las soluciones de nuestra propuesta dominan a las soluciones del NSGA-II y MOPSOcd, y siendo superado por las soluciones del SMS-EMOA.

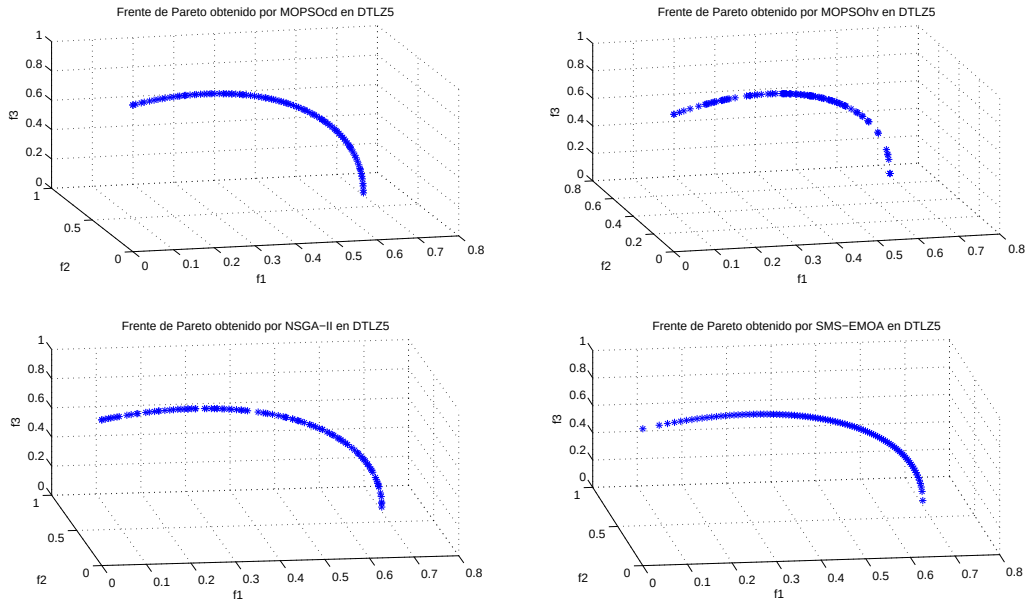

Figura 3.20: Resultados gráficos correspondientes al problema DTLZ5.

	Espaciado			
Algoritmo	**Menor**	**Mayor**	**Promedio**	**Desviación**
MOPSOhv	0.128037	0.220886	**0.158278**	**0.026187**
MOPSOcd	0.271101	0.501514	**0.343938**	**0.052728**
NSGA-II	0.011479	0.029482	**0.020970**	**0.004354**
SMS-EMOA	0.009705	0.020519	**0.013185**	**0.002414**
	DGI			
MOPSOhv	0.007386	0.015926	**0.009931**	**0.002293**
MOPSOcd	0.009901	0.062068	**0.036889**	**0.012547**
NSGA-II	0.000083	0.001194	**0.000648**	**0.000236**
SMS-EMOA	0.010475	0.011392	**0.010786**	**0.000231**
	Hipervolumen			
MOPSOhv	0.000000	0.000616	**0.000391**	**0.000277**
MOPSOcd	0.000000	0.000000	**0.000000**	**0.000000**
NSGA-II	0.303244	0.437448	**0.361529**	**0.028920**
SMS-EMOA	0.282821	0.374890	**0.340557**	**0.024323**
	Cobertura			
Algoritmo	**MOPSOhv**	**MOPSOcd**	**NSGA-II**	**SMS-EMOA**
MOPSOhv	—	**0.920000**	**0.000000**	**0.000000**
MOPSOcd	0.000000	—	0.000000	0.000000
NSGA-II	**0.958000**	0.996000	—	0.479500
SMS-EMOA	**0.630000**	0.640000	0.001500	—

Tabla 3.16: Resultados correspondientes al problema DTLZ6.

El problema DTLZ6 es muy dificil para nuestra propuesta (MOPSOhv) y el MOPSOcd para converger de manera óptima, como se muestra en la tabla 3.16 ya que no logran acerce al frente de Parto, como se muestra en la gráfica 3.21. Sin embargo, nuestra propuesta domina a las soluciones del MOPSOcd. Nuestra propuesta es superada por el SMS-EMOA y NSGA-II y conforme a estos resultados se puede crear una jerarquía en orden descendente en términos de convergencia del conjunto de soluciones próximas al frente de Pareto real:

1. NSGA-II
2. SMS-EMOA
3. MOPSOhv
4. MOPSOcd

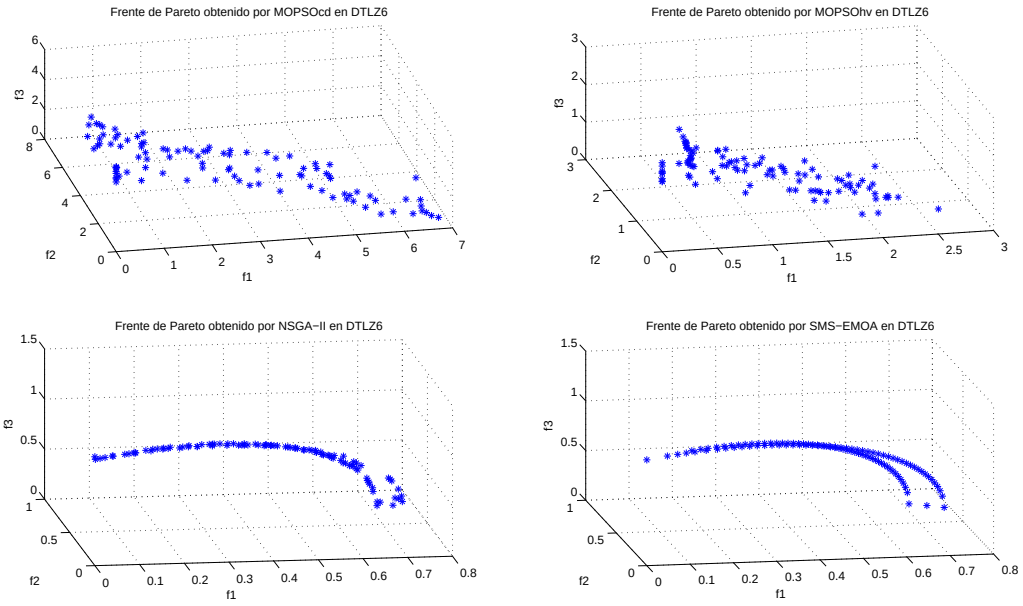

Figura 3.21: Resultados gráficos correspondientes al problema DTLZ6.

	Espaciado			
Algoritmo	**Menor**	**Mayor**	**Promedio**	**Desviación**
MOPSOhv	0.029933	0.140179	**0.085100**	**0.025882**
MOPSOcd	0.063885	0.453648	**0.114527**	**0.112693**
NSGA-II	0.043496	0.085744	**0.068548**	**0.009758**
SMS-EMOA	0.044055	0.063838	**0.057420**	**0.005603**
	DGI			
MOPSOhv	0.001446	0.005356	**0.002800**	**0.001392**
MOPSOcd	0.001228	0.002112	**0.001558**	**0.000205**
NSGA-II	0.000802	0.005285	**0.001123**	**0.000957**
SMS-EMOA	0.013925	0.021469	**0.015064**	**0.002691**
	Hipervolumen			
MOPSOhv	2.359510	2.812829	**2.608814**	**0.123164**
MOPSOcd	2.430821	2.744141	**2.648845**	**0.072522**
NSGA-II	2.578211	3.011594	**2.974631**	**0.091615**
SMS-EMOA	5.055289	5.528637	**5.456793**	**0.168653**
	Cobertura			
Algoritmo	**MOPSOhv**	**MOPSOcd**	**NSGA-II**	**SMS-EMOA**
MOPSOhv	—	**0.460000**	**0.061500**	**0.00000**
MOPSOcd	0.003000	—	0.000000	0.000000
NSGA-II	0.008000	0.658500	—	0.000000
SMS-EMOA	**0.940000**	0.890000	0.980000	—

Tabla 3.17: Resultados correspondientes al problema DTLZ7.

El frente discontinuo de DTLZ7 causa dificultad a nuestra propuesta (MOPSOhv) para converger de manera óptima, como se muestra en la tabla 3.17. En la gráfica 3.22 se muestra que las partículas se concentran en una mayor parte del frente de Pareto. Nuestra propuesta es superada por el SMS-EMOA y NSGA-II y conforme a estos resultados se puede crear una jerarquía en orden descendente en términos de convergencia del conjunto de soluciones próximas al frente de Pareto real:

1. NSGA-II
2. SMS-EMOA
3. MOPSOhv
4. MOPSOcd

También, se observa que en las soluciones de nuestra propuesta dominan a las soluciones del NSGA-II y MOPSOcd, y siendo superado por las soluciones del SMS-EMOA.

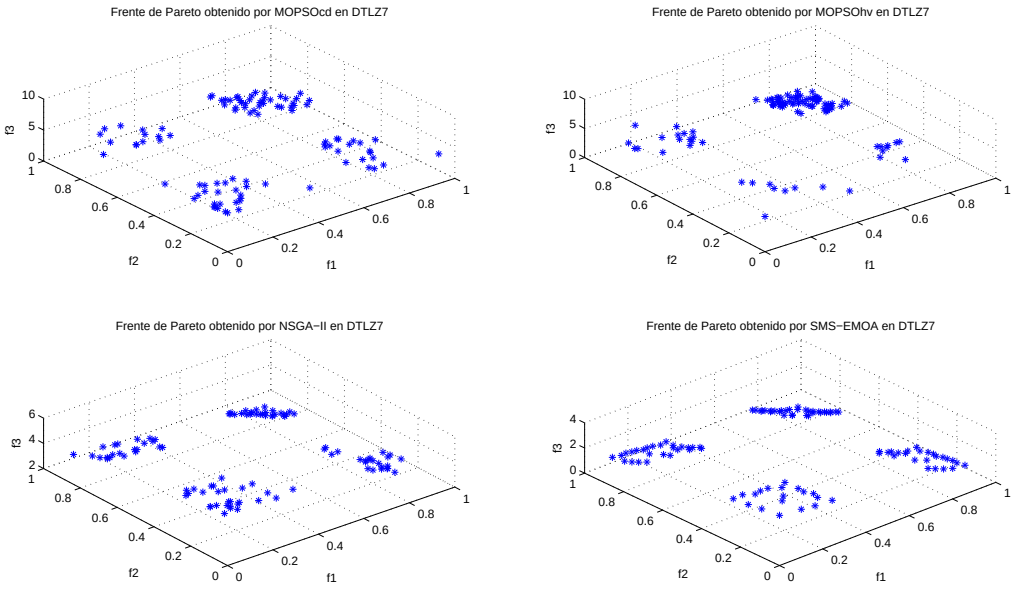

Figura 3.22: Resultados gráficos correspondientes al problema DTLZ7.

3.5. Resultados de escalabilidad

Se utiliza el problema DTLZ2 para probar la escalabilidad de nuestro algoritmo de dos a diez objetivos. Las soluciones son comparadas con el NSGA-II, MOPSO-cd y el SMS-EMOA. Se realizaron diez ejecuciones independientes, con 100 individuos y 1200 generaciones usando los parámetros de las tablas 3.1, 3.2 y 3.3. El punto $\vec{1.1}$ es utlizado como referencia para la métrica del hipervolumen.

	2 Objetivos			
Algoritmo	**Menor**	**Mayor**	**Promedio**	**Desviación**
MOPSOhv	0.006496	0.008450	**0.007529**	**0.000509**
MOPSOcd	0.006508	0.007755	**0.007165**	**0.000350**
NSGA-II	0.006558	0.008114	**0.007298**	**0.000441**
SMS-EMOA	0.007164	0.008134	**0.007687**	**0.000324**
	3 Objetivos			
MOPSOhv	0.040552	0.075311	**0.060766**	**0.010465**
MOPSOcd	0.046830	0.059367	**0.052755**	**0.003508**
NSGA-II	0.047551	0.062545	**0.055653**	**0.004653**
SMS-EMOA	0.040739	0.045160	**0.042945**	**0.001627**
	4 Objetivos			
MOPSOhv	0.060814	0.103256	**0.076769**	**0.013224**
MOPSOcd	0.255004	0.351009	**0.307083**	**0.029160**
NSGA-II	0.099124	0.121004	**0.110590**	**0.007367**
SMS-EMOA	0.060019	0.063409	**0.061934**	**0.001271**
	5 Objetivos			
MOPSOhv	0.062663	0.115829	**0.096073**	**0.016611**
MOPSOcd	0.357735	0.509170	**0.439317**	**0.048635**
NSGA-II	0.176407	0.213365	**0.194858**	**0.012554**
SMS-EMOA	—	—	—	—
	6 Objetivos			
MOPSOhv	0.069909	0.146411	**0.115437**	**0.022606**
MOPSOcd	0.472034	0.601704	**0.536438**	**0.044512**
NSGA-II	0.326463	0.462761	**0.393104**	**0.040260**
SMS-EMOA	—	—	—	—
	7 Objetivos			
MOPSOhv	0.104604	0.198331	**0.134378**	**0.027768**
MOPSOcd	0.567486	0.800344	**0.666199**	**0.064909**
NSGA-II	0.555849	0.750226	**0.626970**	**0.053539**
SMS-EMOA	—	—	—	—
	8 Objetivos			
MOPSOhv	0.091730	0.173281	**0.122761**	**0.025681**
MOPSOcd	0.656487	0.898391	**0.768978**	**0.083783**
NSGA-II	0.707636	0.843402	**0.773139**	**0.048302**
SMS-EMOA	—	—	—	—

	9 Objetivos			
MOPSOhv	0.069848	0.150881	**0.109239**	**0.026673**
MOPSOcd	0.656487	0.898391	**0.768978**	**0.083783**
NSGA-II	0.883374	0.977749	**0.913665**	**0.025785**
SMS-EMOA	—	—	—	—
	10 Objetivos			
MOPSOhv	0.090387	0.180091	**0.129630**	**0.028221**
MOPSOcd	0.682277	0.953287	**0.826685**	**0.093907**
NSGA-II	0.936894	1.132359	**1.032422**	**0.071610**
SMS-EMOA	—	—	—	—

Tabla 3.18: Resultados de la métrica de espaciado para DTLZ2 de 2 a 10 objetivos.

La tabla 3.18 muestra como la distribución de las soluciones mejoran para nuestra propuesta al aumentar el número de objetivos del problema DTLZ2. Mientras que en el NSGA-II y MOPSOcd el valor de la métrica de espaciado crece al aumentar el número de objetivos nuestra propuesta se mantiene constante.

	2 Objetivos			
Algoritmo	**Menor**	**Mayor**	**Promedio**	**Desviación**
MOPSOhv	0.420910	0.420993	**0.420962**	**0.000031**
MOPSOcd	0.420317	0.420437	**0.420372**	**0.000037**
NSGA-II	0.418849	0.419966	**0.419588**	**0.000323**
SMS-EMOA	0.421020	0.421027	**0.421023**	**0.000003**
	3 Objetivos			
MOPSOhv	0.597900	0.697158	**0.644946**	**0.030303**
MOPSOcd	0.622449	0.698677	**0.667862**	**0.024190**
NSGA-II	0.676373	0.708493	**0.697689**	**0.009213**
SMS-EMOA	0.757991	0.758168	**0.758071**	**0.000056**
	4 Objetivos			
MOPSOhv	0.570237	0.735955	**0.649871**	**0.046065**
MOPSOcd	0.000000	0.000000	**0.000000**	**0.000000**
NSGA-II	0.805413	0.876853	**0.834518**	**0.019843**
SMS-EMOA	1.044708	1.044792	**1.044734**	0.000030
	5 Objetivos			
MOPSOhv	0.526935	0.821024	**0.686608**	**0.091850**
MOPSOcd	0.000000	0.000000	**0.000000**	**0.000000**
NSGA-II	0.624795	0.874966	**0.806271**	**0.075088**
SMS-EMOA	—	—	—	—
	6 Objetivos			
MOPSOhv	0.588840	0.897113	**0.768027**	**0.106912**
MOPSOcd	0.000000	0.000000	**0.000000**	**0.000000**
NSGA-II	0.016168	0.385332	**0.195606**	**0.133364**

SMS-EMOA	—	—	—	—
	7 Objetivos			
MOPSOhv	0.779870	0.986701	**0.897171**	**0.059289**
MOPSOcd	0.000000	0.000000	**0.000000**	**0.000000**
NSGA-II	0.000960	0.336963	**0.146724**	**0.119704**
SMS-EMOA	—	—	—	—
	8 Objetivos			
MOPSOhv	0.711453	1.069716	**0.923948**	**0.098939**
MOPSOcd	0.000000	0.000000	**0.000000**	**0.000000**
NSGA-II	0.071442	0.358558	**0.185958**	**0.088435**
SMS-EMOA	—	—	—	—
	9 Objetivos			
MOPSOhv	0.767567	1.063881	**0.894945**	**0.098291**
MOPSOcd	0.000000	0.000000	**0.000000**	**0.000000**
NSGA-II	0.039463	0.397496	**0.212041**	**0.124973**
SMS-EMOA	—	—	—	—
	10 Objetivos			
MOPSOhv	0.946736	1.228404	**1.080952**	**0.085351**
MOPSOcd	0.000000	0.000000	**0.000000**	**0.000000**
NSGA-II	0.015169	0.440330	**0.224003**	**0.127577**
SMS-EMOA	—	—	—	—

Tabla 3.19: Resultados de la métrica de hipervolumen para DTLZ2 con 2 a 10 objetivos.

La tabla 3.19 muestra los resultados obtenidos para la métrica del hipervolumen para el problema DTLZ2 con 2 a 10 objetivos. El NSGA-II muestra un deterioro en la calidad de las soluciones cuando aumenta el número de objetivos del problema con un costo computacional bajo. El SMS-EMOA muestra tener buenos resultados, pero es muy costoso, computacionalmente hablando. MOPSOcd muestra que su desempeño no es escalable ya que se degrada rápidamente al aumentar el número de funciones objetivo.

Conforme a estos resultados se puede crear una jerarquía en orden descendente en términos del valor del hipervolumen del conjunto de soluciones próximas al frente de Pareto real:

1. MOPSOhv
2. NSGA-II
3. SMS-EMOA
4. MOPSOcd

Nuestra propuesta muestra tener buenos resultados en la calidad de las soluciones cuando aumentan el número de objetivos del problema al mantenerse constante en la métrica del hipervolumen y manteniendo un costo computacional razonable, lo que hace a nuestro algoritmo competitivo respecto a los demás como se muestra en la figura 3.23.

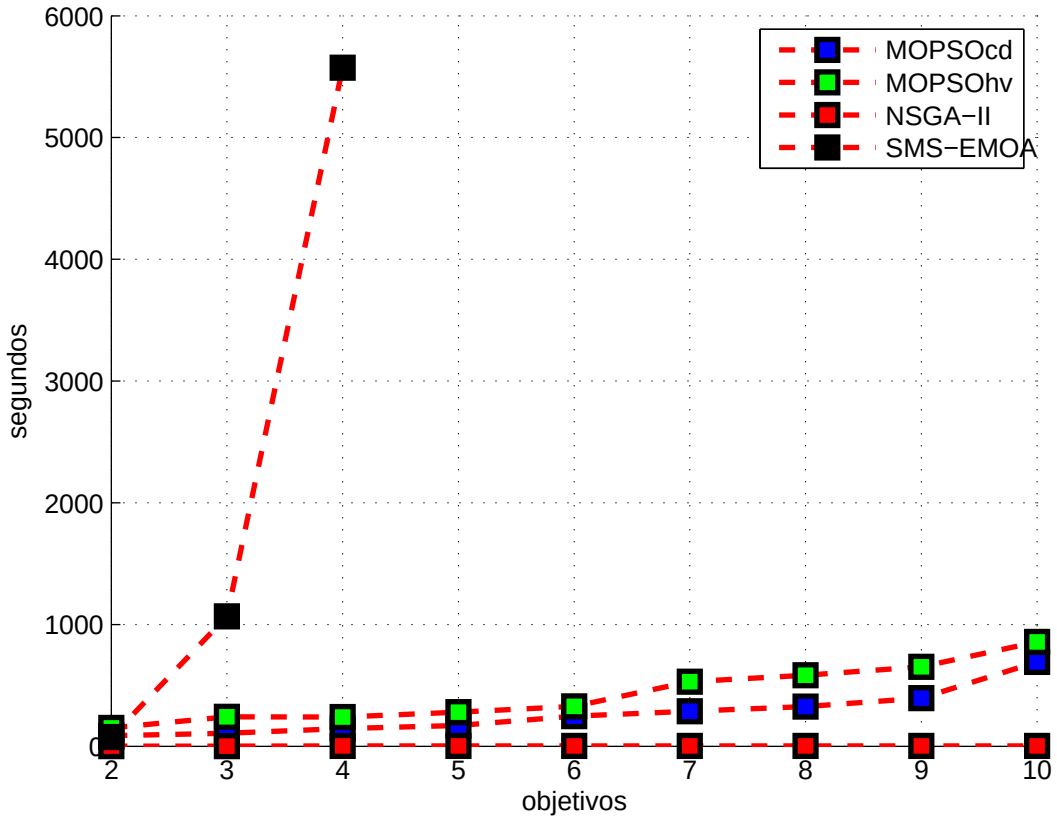

Figura 3.23: Resultados de tiempo en segundos aumentando el número de objetivos del problema DTLZ2.

Capítulo 4

Conclusiones y trabajo futuro

4.1. Resumen

El objetivo principal de este trabajo de tesis fue proponer un nuevo algoritmo multi-objetivo basado los cúmulos de partículas la cual mostrara ser competitiva con respecto a algoritmos evolutivos multiobjetivo (AEMOs) representativos del estado del arte.

Para llevar a cabo este objetivo se realizó un estudio de las metaheurísticas multi-objetivo basadas en cúmulos de partículas que existen actualmente. Posteriormente, se realizó un estudio del denominado hipervolumen para poder incorporarlo en el mecanismo de selección de nuestra metaheurística. A este respecto, se identificaron varias formas de utilizar el hipervolumen como mecanismo de selección, sobresaliendo el uso del algoritmo HypE que usa un método para aproximar las contribuciones al hipervolumen. Como producto de este análisis se propuso un algoritmo de cúmulos de partículas multi-objetivo que aprovecha las contribuciones al hipervolumen para seleccionar la mejor guía global a fin de generar soluciones potenciales no dominadas.

El algoritmo propuesto fue comparado con respecto a NSGA-II, SMS-EMOA y MOPSOcd. La evaluación se realizó utilizando un conjunto de problemas multi-objetivo que reúnen diferentes características que causan dificultades a un algoritmo evolutivo multi-objetivo.

4.2. Conclusiones

A partir de los experimentos realizados se desprenden las siguientes conclusiones:

- Una de las principales dificultades en extender el algoritmo del PSO a una versión multi-objetivo, es encontrar la mejor forma de seleccionar al líder de cada partícula en el cúmulo. Esta dificultad radica en que no hay una noción clara sobre cómo definir quién es el mejor alcanzado hasta el momento (***pBest***) y el mejor alcanzado por toda la población (***gBest***). En los problemas de optimización multi-objetivo, todas las soluciones no dominadas son igualmente buenas, por lo que cualquiera de ellas puede adoptarse como líder. Por tanto, el uso del hipervolumen nos permitió seleccionar al mejor líder alcanzado de un conjunto de soluciones no dominadas. La selección se realiza conforme a su contribución al hipervolumen, es decir, aquella partícula que contribuye más al valor del hipervolumen es seleccionada como líder.
- El uso del hipervolumen para seleccionar al líder en nuestro algoritmo muestra una mejora con respecto al algoritmo MOPSOcd.

- La forma de seleccionar al conjunto de partículas que influyen en el entorno social del algoritmo ($pBest$) mejora considerablemente la búsqueda de nuevas soluciones no dominadas. Es mejor utilizar el archivo o la población secundaria, en lugar de la poblacion primaria, para actualizar el $pBest$ del algoritmo propuesto.

- La contribución al hipervolumen es un valor que se basa en la métrica del hipervolumen. Por lo que, resulta ser muy costoso computacionalmente hablando. Sin embargo, el uso de algoritmos que hacen uso de metodos de muestreo para calcular la contribución al hipervolumen resultan ser más eficientes. El uso de HypE es un algoritmo que utiliza el método de Monte Carlo y muestra ser más eficiente al calcular las contribuciones al hipervolumen.

- El uso del algoritmo de HypE para calcular las contribuciones al hipervolumen permite aumentar el número de generaciones del algoritmo y el número de objetivos del problema.

- Debido a que se utiliza una aproximación de la contribución al hipervolumen, se decidió utilizar un operador de turbulencia para evitar quedar atrapado en frentes de Pareto locales.

- Los resultados obtenidos en promedio para los primeros tres problemas ZDT son similares. Para el problema ZDT4 es mejor que los resultados obtenidos por el SMS-EMOA y marginalmente peores que los del NSGA-II. Así mismo, nuestros resultados son mejores que los de MOPSOcd. Para el problema ZDT6 los resultados son ligeramente mejores que los de SMS-EMOA y ligeramente peores que los de NSGA-II, siendo mucho mejores que los del MOPSOcd.

- Los resultados obtenidos por nuestra propuesta para los problemas DTLZ1, DTLZ3 y DTLZ6 son mejores que los del MOPSOcd. Sin embargo, nuestro algoritmo no pudo obtener buenas aproximaciones al frente de Pareto real, y nuestros resultados fueron dominados por los de SMS-EMOA y NSGA-II. Los resultados para los demás problemas (DTLZ2, DTLZ4, DTLZ5 y DTLZ7) muestran que en la mayoria de los casos, nuestra propuesta obtiene soluciones que dominan a las de los algoritmos NSGA-II, SMS-EMOA y MOPSOcd.

- Los resultados obtenidos de nuestra propuesta muestran que en la mayoria de los problemas (2 y 3 objetivos), en promedio ,es competitiva con respecto a los algoritmos NSGA-II, SMS-EMOA y MOPSOcd.

- Nuestro algoritmo mantiene un buen desempeño al aumentar el número de objetivos del problema, manteniendo un costo computacional razonable. Esto hace que nuestro algoritmo sea competitivo con respecto a los algoritmos NSGA-II, al SMS-EMOA y al MOPSOcd.

4.3. Trabajo futuro

Existen diversas formas de poder diseñar un algoritmo multi-objetivo con base en la metaheurística de cúmulos de partículas, pues es posible usar diferentes tipos de topologías

o conexiones para que las partículas interactúen o influyan entre ellas a fin de generar buenas aproximaciones hacia el verdadero frente de Pareto.

En particular se podrían realizar las siguientes extensiones a nuestro algoritmo:

- Utilizar modelos de configuración diferentes al modelo completo que se usó aquí (modelo cognitivo, modelo social o modelo social exclusivo).
- Utilizar otros apectos avanzados que pueden acelerar la convergencia del algoritmo (factor de constricción, factor de inercia adaptable o control de velocidad).
- Seleccionar a los guías locales de otra manera, de tal forma, que afecten la parte cognitiva del algoritmo permitiendo generar nuevas soluciones.
- Seleccionar un mejor estimador de densidad para hacer el reemplazo de los soluciones no dominadas en la población secundaria.

Bibliografía

[Angeline et al., 1999] Angeline, P. J., Michalewicz, Z., Schoenauer, M., Yao, X., and Zalzala, A., editors (1999). *The Pareto Archived Evolution Strategy: A New Baseline Algorithm for Pareto Multiobjective Optimisation*, volume 1, Mayflower Hotel, Washington D.C., USA. IEEE Press.

[Bäck et al., 1997] Bäck, T., Fogel, D., and Michalewicz, Z. (1997). *Handbook of Evolutionary Computation*. Oxford Univ. Press.

[Bader et al., 2008] Bader, J., Deb, K., and Zitzler, E. (2008). Faster Hypervolume-Based Search Using Monte Carlo Sampling. In Ehrgott, M., Naujoks, B., Stewart, T. J., and Wallenius, J., editors, *MCDM*, volume 634 of *Lecture Notes in Economics and Mathematical Systems*, pages 313–326. Springer.

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

[Beume, 2009] Beume, N. (2009). S-Metric Calculation by Considering Dominated Hypervolume as Klee's Measure Problem. *Evolutionary Computation*, 17(4):477–492.

[Beume et al., 2009] Beume, N., Fonseca, C. M., López-Ibáñez, M., Paquete, L., and Vahrenhold, J. (2009). On the Complexity of Computing the Hypervolume Indicator. *IEEE Trans. Evolutionary Computation*, 13(5):1075–1082.

[Beume et al., 2007] Beume, N., Naujoks, B., and Emmerich, M. (2007). SMS-EMOA: Multiobjective Selection Based on Dominated Hypervolume. *European Journal of Operational Research*, 181(3):1653–1669.

[Braberman et al., 2007] Braberman, V. A., Obes, J. L., Olivero, A., and Schapachnik, F. (2007). Hypervolume Approximation in Timed Automata Model Checking. In Raskin, J.-F. and Thiagarajan, P. S., editors, *FORMATS*, volume 4763 of *Lecture Notes in Computer Science*, pages 69–81. Springer.

[Bringmann and Friedrich, 2008] Bringmann, K. and Friedrich, T. (2008). Approximating the volume of unions and intersections of high-dimensional geometric objects. In Hong, S.-H., Nagamochi, H., and Fukunaga, T., editors, *ISAAC*, volume 5369 of *Lecture Notes in Computer Science*, pages 436–447. Springer.

[Bringmann and Friedrich, 2009] Bringmann, K. and Friedrich, T. (2009). Don't be Greedy when Calculating Hypervolume Contributions. In Garibay, I. I., Jansen, T., Wiegand, R. P., and Wu, A. S., editors, *FOGA*, pages 103–112. ACM.

[Bringmann and Friedrich, 2010] Bringmann, K. and Friedrich, T. (2010). An Efficient Algorithm for Computing Hypervolume Contributions. *Evolutionary Computation*, 18(3):383–402.

[Cagnina, 2010] Cagnina, L. C. (2010). *Optimización Mono y Multiobjetivo a través de una Heurística de Interligencia Colectiva*. PhD thesis, Universidad Nacional de San Luis.

[Chang et al., 2005] Chang, J.-F., Chu, S.-C., Roddick, J. F., and Pan, J.-S. (2005). A Parallel Particle Swarm Optimization Algorithm with Communication Strategies. *J. Inf. Sci. Eng.*, 21(4):809–818.

[Clerc, 1999] Clerc, M. (1999). The Swarm and the Queen: Towards a Deterministic and Adaptive Particle Swarm Optimization. In *Congress on Evolutionary Computation*, volume 3, pages 1951–1957.

[Clerc and Kennedy, 2002] Clerc, M. and Kennedy, J. (2002). The Particle Swarm: Explosion, Stability and Convergence in a Multidimensional Complex Space. *IEEE Trans. Evolutionary Computation*, 6(1):58–73.

[Coello et al., 2006] Coello, C. A. C., Lamont, G. B., and Veldhuizen, D. A. V. (2006). *Evolutionary Algorithms for Solving Multi-Objective Problems (Genetic and Evolutionary Computation)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA.

[Coello et al., 2004] Coello, C. A. C., Pulido, G. T., and Lechuga, M. S. (2004). Handling Multiple Objectives With Particle Swarm Optimization. *IEEE Trans. Evolutionary Computation*, 8(3):256–279.

[Darwin, 1959] Darwin, C. (1959). *The origin of species by means of natural selection by Means of Natural Selection, or The Preservation of Struggle for Life*. Penguin Book Ltd. New York :Hurst,. http://www.biodiversitylibrary.org/bibliography/2106.

[Deb et al., 2000] Deb, K., Agrawal, S., Pratap, A., and Meyarivan, T. (2000). A Fast Elitist Non-dominated Sorting Genetic Algorithm for Multi-objective Optimisation: NSGA-II. In Schoenauer, M., Deb, K., Rudolph, G., Yao, X., Lutton, E., Guervós, J. J. M., and Schwefel, H.-P., editors, *PPSN*, volume 1917 of *Lecture Notes in Computer Science*, pages 849–858. Springer.

[Deb et al., 2002] Deb, K., Thiele, L., Laumanns, M., and Zitzler, E. (2002). Scalable Multi-Objective Optimization Test Problems. In *Congress on Evolutionary Computation (CEC 2002)*, pages 825–830. IEEE Press.

[Eberhart et al., 1996] Eberhart, R., Simpson, P., and Dobbins, R. (1996). *Computational intelligence PC tools*. Academic Press Professional, Inc., San Diego, CA, USA.

[Eberhart et al., 2001] Eberhart, R. C., Shi, Y., and Kennedy, J. (2001). *Swarm Intelligence (The Morgan Kaufmann Series in Evolutionary Computation)*. Morgan Kaufmann, 1st edition.

[Emmerich et al., 2005] Emmerich, M., Beume, N., and Naujoks, B. (2005). An EMO Algorithm Using the Hypervolume Measure as Selection Criterion. In Coello, C. A. C., Aguirre, A. H., and Zitzler, E., editors, *EMO*, volume 3410 of *Lecture Notes in Computer Science*, pages 62–76. Springer.

[Everson et al., 2002] Everson, R. M., Fieldsend, J. E., and Singh, S. (2002). Full Elite Sets for Multi-Objective Optimisation. Conference on Adaptive Computing in Design and Manufacture, pages 343–354. In I. Parmee (Eds.).

[Fogel, 1964] Fogel, L. (1964). *On the Organization of Intellect*. University of California, Los Angeles - Engineering.

[Fonseca et al., 2006] Fonseca, C. M., Paquete, L., and López-Ibáñez, M. (2006). An Improved Dimension-Sweep Algorithm for the Hypervolume Indicator. In *Proceedings of the 2006 Congress on Evolutionary Computation (CEC 2006)*, pages 1157–1163. IEEE Press, Piscataway, NJ.

[Fu et al., 2011] Fu, W., Johnston, M., and Zhang, M. (2011). Hybrid Particle Swarm Optimisation based on history Information Sharing. In Krasnogor, N. and Lanzi, P. L., editors, *GECCO*, pages 77–84. ACM.

[Hoffman, 1989] Hoffman, A. (1989). Arguments on evolution. A paleontologist's perspective. *American Journal of Physical Anthropology, Oxford University Press USA*, 80(3):405–406.

[Holland, 1992] Holland, J. H. (1992). *Adaptation in Natural and Artificial Systems*. MIT Press, Cambridge, MA, USA.

[Hu and Eberhart, 2002] Hu, X. and Eberhart, R. (2002). Multiobjective Optimization Using Dynamic Neighborhood Particle Swarm Optimization. *Computational Intelligence, Proceedings of the World on Congress on*, 2:1677–1681.

[Hu et al., 2003] Hu, X., Eberhart, R., and Shi, Y. (2003). Particle Swarm with Extended Memory for Multiobjective Optimization. In *Swarm Intelligence Symposium, 2003. SIS '03. Proceedings of the 2003 IEEE*, pages 193 – 197.

[Kennedy, 1997] Kennedy, J. (1997). The Particle Swarm: Social Adaptation of Knowledge. In *Proceedings of IEEE International Conference on Evolutionary Computation*, pages 303–308.

[Kennedy and Eberhart, 1995] Kennedy, J. and Eberhart, R. (1995). Particle Swarm Optimization. *IEEE International Conference on Neural Networks. Proceedings.*, 4:1942–1948 vol.4.

[Kennedy and Mendes, 2006] Kennedy, J. and Mendes, R. (2006). Neighborhood topologies in fully informed and Best of Neighborhood Particle Swarms. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews.*, 36(4):515–519.

[Lechuga and Rowe, 2005] Lechuga, M. S. and Rowe, J. E. (2005). Particle Swarm Optimization and Fitness Sharing to Solve Multiobjective Optimization Problems. In *Congress on Evolutionary Computation*, pages 1204–1211. IEEE.

[Martínez and Coello, 2011] Martínez, S. Z. and Coello, C. A. C. (2011). A Multi-objective Particle Swarm Optimizer Based on Decomposition. In Krasnogor, N. and Lanzi, P. L., editors, *GECCO*, pages 69–76. ACM.

[Mendel et al., 1965] Mendel, G., Fisher, R. A., and Bennett, J. H. (1965). *Experiments in plant hybridisation: Mendel's original paper in English translation, with commentary and assessment by the late Sir Ronald A. Fisher, together with a reprint of W. Bateson's biographical notice of Mendel / Gregor Mendel ; edited by J.H. Bennett* . Oliver and Boyd, Edinburgh.

[Miettinen, 1999] Miettinen, K. (1999). *Nonlinear Multiobjective Optimization*, volume 12 of *International Series in Operations Research and Management Science*. Kluwer Academic Publishers, Dordrecht.

[Mostaghim et al., 2007] Mostaghim, S., Branke, J., and Schmeck, H. (2007). Multiobjective Particle Swarm Optimization on Computer Grids. In Lipson, H., editor, *GECCO*, pages 869–875. ACM.

[Mostaghim and Teich, 2004] Mostaghim, S. and Teich, J. (2004). Covering Pareto-optimal Fronts by Subswarms in Multi-objective Particle Swarm Optimization. In *IEEE Proceedings, World Congress on Computational Intelligence (CEC'04)*, pages 1404–1411, Portland, USA.

[Osyczka, 1985] Osyczka, A. (1985). *Multicriterion Optimization in Engineering with FORTRAN Programs*. Academic Press.

[Padhye, 2009] Padhye, N. (2009). Comparison of Archiving Methods in Multiobjective Particle Swarm Optimization (MOPSO): Empirical Study. In Rothlauf, F., editor, *GECCO*, pages 1755–1756. ACM.

[Parsopoulos and Vrahatis, 2002] Parsopoulos, K. E. and Vrahatis, M. N. (2002). Particle Swarm Optimization Method in Multiobjective Problems. In *SAC*, pages 603–607. ACM.

[Raquel and Jr., 2005] Raquel, C. R. and Jr., P. C. N. (2005). An Effective use of Crowding Distance in Multiobjective Particle Swarm Optimization. In Beyer, H.-G. and O'Reilly, U.-M., editors, *GECCO*, pages 257–264. ACM.

[Reehuis, 2010] Reehuis, E. (2010). Multiobjective Robust Optimization of Water Distribution Networks. Master's thesis, Leiden Institute of Advanced Computer Science, Leiden University, The Netherlands.

[Reyes Sierra and Coello Coello, 2006] Reyes Sierra, M. and Coello Coello, C. A. (2006). Multi-Objective Particle Swarm Optimizers: A Survey of the State of the Art. *International Journal of Computational Intelligence Research*, 2(3):287–308.

[Rodríguez Villalobos, 2011] Rodríguez Villalobos, C. A. (2011). A New Multiobjective Evolutionary Algorithm Based on a Performance Assessment Indicator. Master's thesis, Centro de Investigación y de Estudios Avanzados del Instituto Politécnico Nacional, campus Zacatenco, Departamento de Ciencias de la Computación, D.F, México.

[Rosenberg, 1967] Rosenberg, R. (1967). *Simulation of Genetic Populations with Biochemical Properties*. PhD thesis, Univ. Michigan.

[Santana Quintero and Coello Coello, 2006] Santana Quintero, L. V. and Coello Coello, C. A. (2006). Una introducción a la Computación Evolutiva y Algunas de sus Aplicaciones en Economía y Finanzas. *Revista de Métodos Cuantitativos para la Economía y la Empresa*, 2:3–26.

[Schaffer, 1985] Schaffer, J. D. (1985). Multiple Objective Optimization with Vector Evaluated Genetic Algorithms. In Grefenstette, J. J., editor, *ICGA*, pages 93–100. Lawrence Erlbaum Associates.

[Shi and Eberhart, 1998a] Shi, Y. and Eberhart, R. (1998a). A Modified Particle Swarm Optimizer. *Evolutionary Computation Proceedings, 1998. IEEE World Congress on Computational Intelligence., The 1998 IEEE International Conference on*, pages 69–73.

[Shi and Eberhart, 1998b] Shi, Y. and Eberhart, R. C. (1998b). *Parameter Selection in Particle Swarm Optimization*, volume 160. Springer.

[Srinivas and Deb, 1994] Srinivas, N. and Deb, K. (1994). Multiobjective optimization using nondominated sorting in genetic algorithms. *Evolutionary Computation*, 2(3):221–248.

[Veldhuizen and Lamont, 1998] Veldhuizen, D. A. V. and Lamont, G. B. (1998). Multiobjective Evolutionary Algorithm Research: A History and Analysis. Technical Report TR-98-03, Department of Electrical and Computer Engineering, Graduate School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, Ohio.

[Weismann, 1893] Weismann, A. (1893). *The germ-plasm : a theory of heredity / by A. Weismann ; translated by W. Newton Parker and H. Ronnfeldt* . W. Scott, London :.

[Yang and Ding, 2007] Yang, Q. and Ding, S. (2007). Novel Algorithm to Calculate Hypervolume Indicator of Pareto Approximation Set. *CoRR*, abs/0704.1196.

[Yang, 2008] Yang, X.-S. (2008). *Nature-Inspired Metaheuristic Algorithms*. Luniver Press.

[Zitzler et al., 2006] Zitzler, E., Brockhoff, D., and Thiele, L. (2006). The Hypervolume Indicator Revisited: On the Design of Pareto-compliant Indicators Via Weighted Integration. In Obayashi, S., Deb, K., Poloni, C., Hiroyasu, T., and Murata, T., editors, *EMO*, volume 4403 of *Lecture Notes in Computer Science*, pages 862–876. Springer.

[Zitzler et al., 2000] Zitzler, E., Deb, K., and Thiele, L. (2000). Comparison of Multiobjective Evolutionary Algorithms: Empirical Results. *Evolutionary Computation*, 8(2):173–195.

[Zitzler et al., 2002] Zitzler, E., Laumanns, M., and Thiele, L. (2002). SPEA2: Improving the Strength Pareto Evolutionary Algorithm for Multiobjective Optimization. In Giannakoglou, K. et al., editors, *Evolutionary Methods for Design, Optimisation and Control with Application to Industrial Problems (EUROGEN 2001)*, pages 95–100. International Center for Numerical Methods in Engineering (CIMNE).

[Zitzler and Thiele, 1998] Zitzler, E. and Thiele, L. (1998). Multiobjective Optimization Using Evolutionary Algorithms: A Comparative Case Study. In Eiben, A. E., Bäck, T., Schoenauer, M., and Schwefel, H.-P., editors, *PPSN*, volume 1498 of *Lecture Notes in Computer Science*, pages 292–304. Springer.

[Zitzler and Thiele, 1999] Zitzler, E. and Thiele, L. (1999). Multiobjective Evolutionary Algorithms: a Comparative Case Study and the Strength Pareto Approach. *IEEE Trans. Evolutionary Computation*, 3(4):257–271.