“Equipo de Algoritmos Evolutivos Multiobjetivo Paralelos”
(*Team Algorithm of pMOEAs*)

por
José Manuel Fernandez Giangreco

Proyecto Final de Tesis

Orientador:
D. Sc. Ing. Benjamín Barán

Ingeniería Informática
Facultad de Ciencias y Tecnología
Universidad Católica
“Nuestra Señora de la Asunción”

Asunción – Paraguay
Enero del 2005

Este trabajo está dedicado a mis padres , a mis hermanos y al amor de mi vida.

Agradecimientos

Al profesor Benjamín Barán, modelo de profesional y por sobre todo, excelente persona. Sin su constante apoyo y orientación este proyecto no hubiera llegado a buen puerto. Su excepcional capacidad de superación frente a los problemas e inigualable visión positiva han inyectado vida y dinamismo al quehacer diario de las personas que tenemos la gran suerte de pertenecer a su grupo de investigación.

A la Profesora Blanca Troche de Trevisán, ex Directora del Centro Nacional de Computación, por haber hecho posible que en dicha institución puedan ser llevados a cabo trabajos de investigación de esta naturaleza. Con personas como ella sin dudas nuestro querido país tendrá un futuro luminoso, de pleno desarrollo y madurez.

A los queridos compañeros del Centro Nacional de Computación, en las personas de Christian Von Lucken, Diego Pinto, Pedro Gardel, Francisco Talavera, Osvaldo Gómez, Diana Benítez. Deseo agradecerles especialmente por su constante apoyo y colaboración en las diferentes instancias del desarrollo del presente trabajo, así también por el excelente ambiente de trabajo del que me han permitido ser parte.

A Felipe Stuardo y Diego Dasso por la colaboración prestada en los laboratorios de la Facultad.

Finalmente una mención especial a todos mis profesores, compañeros y amigos de la Facultad de Ciencias y Tecnología, con quienes he compartido estos años de formación. Mi deseo es que el conocimiento y la experiencia que hemos adquirido a través de nuestro paso por esta facultad nos sirvan para forjarnos una vida digna, honesta y plena, al servicio de nuestras familias y de nuestro país.

Índice General

Índice de Figuras

Índice de Tablas

Lista de Acrónimos

A-Team	*Asynchronous Team*
CNSGA2	*Controlled Elitist* NSGA-II
COW	*Cluster Of Workstations*
CPU	*Control Process Unit*
DM	*decision maker*
E	*Razón de Error*
EA	*Evolutionary Algorithm*
EMOO	*Evolutionary MultiObjective Optimization*
G	Distancia generacional
GA	*Genetic Algorithms*
GRVND	Generación real de vectores no dominados
GVND	Generación de vectores no dominados
LAN	*Local Area Network*
MOEA	*MultiObjective Evolutionary Algorithm*
MOGA	*Multiobjective Genetic Algorithm*
MOP	*Multiobjective Optimization Problem*
NPGA	*Niched Pareto Genetic Algorithm*
NSGA	*Nondominated Sorting Genetic Algorithm*
NSGA2	*Nondominated Sorting Genetic Algorithm II*
pMOEA	*parallel Multiobjective Evolutionary Algorithm*
RGVND	Razón de generación de vectores no dominados
RW	*Roulette-Whell selection*
SOP	*Single Objective Problem*
SPEA	*Strength Pareto Evolutionary Algorithm*
SPEA2	*Strength Pareto Evolutionary AlgorithmII*
TA	*Team Algorithm*
TA-MOEA E	*Team Algoritm of MOEAs Elitist*

TA-MOEA P	*Team Algoritm of MOEAs Probabilistic*
TA-MOEA S	*Team Algoritm of MOEAs Static*
TA-MOEA	*Team Algorithm MultiObjetive Evolutionary Algorithm*
TS	*Tournament Selection*

Capítulo 1

Introducción

En la búsqueda de soluciones a problemas del mundo real puede ser necesario satisfacer de manera simultánea múltiples objetivos, los cuales pueden ser contradictorios [MIE01]. De existir la posibilidad de combinar los diferentes objetivos de un problema y conociendo la mejor manera de hacerlo, entonces se puede considerar la existencia de un único objetivo a optimizar. En este caso, para obtener la solución óptima del problema basta con encontrar el mínimo o el máximo de una única función que resume todos los objetivos. Sin embargo, lo usual es que no se conozca de que manera combinar los diferentes objetivos, o esto sea inadecuado, cuando no imposible. Entonces, se dice que el problema es un Problema de Optimización Multiobjetivo (*Multiobjective Optimization Problem* - MOP) [MIE01, GOL89, COE01, VEL99].

En los problemas de optimización multiobjetivo con objetivos contradictorios no existe una solución única que pueda ser considerada la mejor, sino un conjunto de soluciones que representan los mejores compromisos entre los objetivos [MIE01]. Éste es un conjunto de soluciones óptimas en el sentido que cada una es mejor que las otras en algún objetivo, pero ninguna es mejor que otra en todos los objetivos simultáneamente. Dicho conjunto es llamado conjunto de soluciones Pareto-óptimas y sus correspondientes vectores en el espacio objetivo constituyen el denominado Frente Pareto [GOL89].

Los Algoritmos Evolutivos (*Evolutionary Algorithms* - EAs) [BAC97] han demostrado ser especialmente adecuados para la optimización multiobjetivo [COE99, ZIT01]. Desde la década de los 80s se han desarrollado varios EAs, capaces de buscar múltiples soluciones Pareto-óptimas en forma simultánea en una sola corrida [COE98, COE99, DEB99, FON93]. Con el paso de los años, la Optimización Evolutiva Multiobjetivo (EMOO - *Evolutionary MultiObjective Optimization*) se ha establecido como un área importante de investigación, recibiendo cada vez mayor atención [VEL99].

En la actualidad, existe un gran número de Algoritmos Evolutivos para Optimización Multiobjetivo (*Multiobjective Evolutionary Algorithms* - MOEA) [COE02]. Con el uso cada vez más

extendido de estos algoritmos en problemas reales de optimización, se hace necesario mejorar el desempeño de los mismos [BAR01, DUA01]. Por lo tanto, para asegurar la aplicabilidad de la técnica de optimización evolutiva multiobjetivo a problemas de complejidad creciente, es necesario mejorar tanto en la efectividad como en la eficiencia de los métodos evolutivos. Para ello, una alternativa es la incorporación de conceptos de paralelismo al diseño de estos algoritmos [DUA01, VEL02].

Los conceptos de paralelismo han estado presentes en la literatura referente a EAs prácticamente desde sus origines. En [CAN98, CAN99a, CAN99b] se presenta una revisión de varios modelos de paralelización para algoritmos evolutivos mono-objetivo. Estos modelos apenas han sido integrados al campo de la optimización evolutiva multiobjetivo [VEL02, VEL03]. Siendo el área de paralelización de algoritmos evolutivos multiobjetivo un área de gran importancia y reciente interés, es necesario determinar las ventajas y desventajas existentes en el desarrollo y aplicación de algoritmos evolutivos multiobjetivo paralelos (*parallel Multiobjective Evolutionary Algorithms* - pMOEAs) [VEL02, VEL03].

Por otra parte tenemos a los TA (*Team Algorithm*), que han demostrado ser una excelente técnica computacional para combinar una variedad de algoritmos corriendo en diferentes procesadores de una red típicamente asíncrona, como las actuales redes de área local conformadas por computadores personales heterogéneos [BAR01, BAR99, BAR96, BAR98, BAR95a, BAR95b].

Ya que se dispone de diversos algoritmos pMOEAs, surgen los *Team Algorithm Multiobjetive Evolutionary Algorithms* (TA-MOEAs) como una alternativa válida para mejorar tanto la efectividad como en la eficiencia al poder explorar con características diferentes porciones, posiblemente disjuntas, del espacio de búsqueda.

A fin de determinar la efectividad de la técnica, como en [GOL89, ZIT99, VEL99], el presente trabajo realiza una comparación de distintos pMOEAs y los TA-MOEAs utilizando diferentes métricas experimentales propuestas para medir su desempeño [DUA00, VEL99]. Para ello, se han realizado implementaciones paralelas de siete algoritmos evolutivos para optimización multiobjetivo y combinando los mismos en tres *Team Algorithm*, uno estático y dos dinámicos, aplicados a diferentes problemas de prueba de distinta dificultad [ZIT99].

En consecuencia, los objetivos propuestos para este trabajo son:

- Consolidar los conceptos más importantes de EMOO.

- Presentar una comparación experimental entre diferentes pMOEAs y el Equipo de pMOEAs (en adelante TA-MOEA).
- Mejorar la capacidad de búsqueda de distintos MOEAs utilizando conceptos de *Team Algorithm*.
- Desarrollar fundamentos para decidir si un TA-MOEA representa una mejor opción con respecto a los pMOEAS aplicados por separado.

El presente trabajo está organizado de acuerdo al siguiente esquema:

- El capítulo 2 da la definición formal de un Problema de Optimización Multiobjetivo y términos relevantes referentes a dichos problemas, además se enuncian los seis problemas utilizados como *test bed*.
- El capítulo 3 presenta los conceptos de computación evolutiva utilizados, haciendo especial referencia a los algoritmos genéticos, sus conceptos, las características que los hacen más atractivos que otros métodos para la resolución de problemas, su funcionamiento, y las razones de su éxito. Se describen los siete algoritmos evolutivos multiobjetivo utilizados en este trabajo.
- El capítulo 4 exhibe conceptos y una breve reseña histórica de los *team algorithms*. Se ilustra el modelo utilizado para la implementación de los TA-MOEAs y se describen en detalle las tres versiones utilizadas.
- El capítulo 5 muestra las distintas métricas utilizadas, así como los resultados obtenidos y las comparaciones de los resultados obtenidos por los siete pMOEAs y los tres TA-MOEAs implementados para la resolución de los seis problemas propuestos.
- Finalmente se presentan las conclusiones y los trabajos futuros en el capítulo 6.

Capítulo 2

Optimización Multiobjetivo

Este capítulo comienza con los conceptos básicos de optimización multiobjetivo, siendo el principal la dominancia Pareto. En la sección 2.2 se definen formalmente el problema de optimización multiobjetivo y los conceptos previamente presentados.

La sección 2.3 está dedicada a los seis problemas ZDT, estos constituyen el conjunto *test bed* de este trabajo. Dando la definición formal y graficando el frente pareto de cada uno de ellos se explican las características y dificultades propias que poseen.

Se ilustra en la sección 2.4 el proceso de resolución de un MOP, el cual se divide en dos subprocesos: de búsqueda y de toma de decisiones, para este último se disponen de métodos a priori, a posteriori e interactivos.

Se finaliza con una pequeña crítica a los métodos tradicionales utilizados para la resolución de MOP, mostrando sus desventajas.

2.1. Conceptos Básicos

A modo de clarificar conceptos se puede considerar sin pérdida de generalidad un problema arbitrario de optimización con $k > 1$ objetivos de igual importancia a ser todos maximizados y en donde no se dispone de conocimiento adicional sobre el problema.

Una solución a este problema puede ser descripta en términos de un vector de decisión $\boldsymbol{x} = (x_1, x_2, \ldots, x_n)$ en el espacio de decisión $\boldsymbol{X}$. Una función $\boldsymbol{f}: \boldsymbol{X} \rightarrow \boldsymbol{Y}$ evalúa la calidad de una solución específica asignándole un vector objetivo $\boldsymbol{y} = (y_1, y_2, \ldots, y_k)$ en el espacio objetivo $\boldsymbol{Y}$ [ZIT04].

Estimar cual es mejor entre dos soluciones $\boldsymbol{x_1}$ y $\boldsymbol{x_2}$ puede ser una tarea compleja. Para poder comparar $\boldsymbol{x_1}$ y $\boldsymbol{x_2}$ se introduce el concepto de dominancia Pareto, el cual establece que un vector

objetivo y_1 domina a otro y_2 si ningún componente de y_1 es peor que el correspondiente componente de y_2 e y_1 es estrictamente mejor en al menos un componente.

A partir del concepto de dominancia Pareto se determina la relación entre x_1 y x_2. Entonces se tienen tres casos posibles [ZIT04]:

1. x_1 domina a x_2 si y_1 domina a y_2.
2. x_2 domina a x_1 si y_2 domina a y_1.
3. x_1 y x_2 son no comparables en cualquier otro caso.

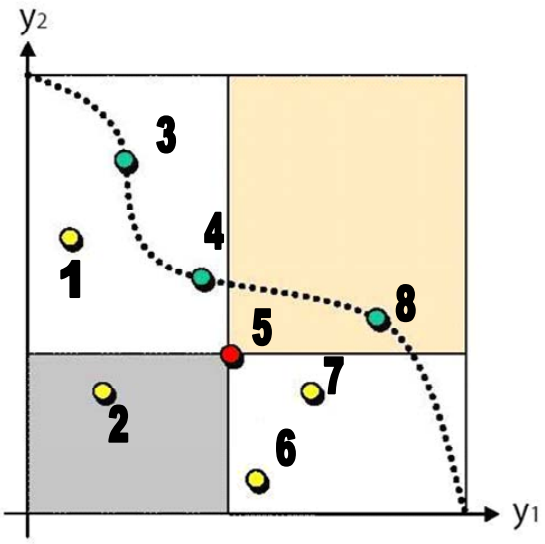

Figura 1. Dominancia Pareto

En la Figura 1 tenemos que para el vector objetivo identificado con el número 5 y de color rojo, en el área inferior izquierda de color lila se encuentran los vectores que domina (vector 2) y en el sector superior derecho de color rosado los que lo dominan (vector 8), en las regiones en blanco se localizan los vectores no comparables (vectores 1, 3, 4, 6 y 7).

Las soluciones óptimas son las que no son dominadas por ninguna otra solución pudiendo existir varias que representan diferentes compromisos entre los objetivos.

Extendiendo el procedimiento ilustrado en la Figura 1 para las 8 soluciones representadas, se llega a clasificar las soluciones en dominadas (vectores 1, 2, 5, 6 y 7) y no dominadas (vectores 3, 4 y 8). En la Figura 2 se diferencian las mismas en colores verde y amarillo.

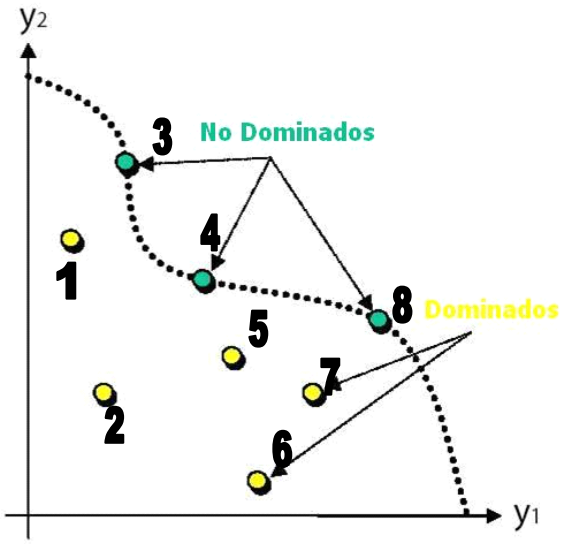

Figura 2. Vectores Objetivos: dominados y no dominados

El conjunto de soluciones óptimas en el espacio de decisión ***X*** es denotado como el Conjunto Pareto y su imagen en el espacio objetivo ***Y*** se denomina el Frente Pareto. El conocimiento acerca de este conjunto ayuda a elegir la mejor solución de compromiso entre los objetivos, esto se logra mediante la búsqueda en el espacio decisión y la evaluación en el espacio objetivo [ZIT04]. Este proceso se grafica en la Figura 3.

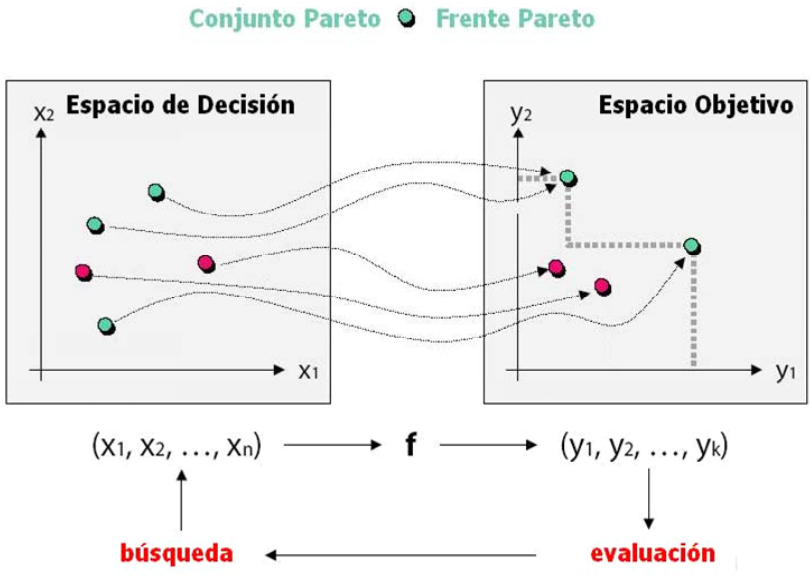

Figura 3. Espacios Decisión y Objetivo

Aunque existan diferentes maneras de tratar un problema de optimización multiobjetivo, por ejemplo, uniendo los diferentes objetivos en una sola función parametrizada, la mayoría de los trabajos en el área de optimización evolutiva multiobjetivo se concentra en la aproximación al Conjunto Pareto [ZIT04]. Por lo tanto se asume en este trabajo que el objetivo de la optimización es encontrar o aproximarse al conjunto de soluciones no dominadas, o sea al Conjunto Pareto.

2.2. Formulación Matemática

Debido a la naturaleza aún incipiente del ámbito de investigación en el área de optimización multiobjetivo, no existe una notación estándar, y no ha sido sino hasta hace muy poco tiempo atrás que los investigadores han empezado a preocuparse de definir con claridad estos aspectos. Sin embargo, en gran parte de los trabajos consultados se percibe aún bastante confusión al respecto. Por lo tanto, es esencial establecer una notación clara antes de iniciar la discusión. A continuación se da la definición formal de un problema de optimización multiobjetivo.

Definición 1: Problema de Optimización Multiobjetivo (*Multiobjective Optimization Problem:* MOP). Un MOP general incluye un conjunto de *n* parámetros (variables de decisión), un conjunto de *k* funciones objetivo, y un conjunto de *m* restricciones. Las funciones objetivo y las restricciones son funciones de las variables de decisión. Luego, el MOP puede expresarse como:

$$\begin{aligned} &\textit{Optimizar} && \boldsymbol{y} = \boldsymbol{f}(\boldsymbol{x}) = (f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), \ldots, f_k(\boldsymbol{x})) \\ &\textit{sujeto a} && \boldsymbol{e}(\boldsymbol{x}) = (e_1(\boldsymbol{x}), e_2(\boldsymbol{x}), \ldots, e_m(\boldsymbol{x})) \geq \boldsymbol{0} \\ &\textit{donde} && \boldsymbol{x} = (x_1, x_2, \ldots, x_n) \in \boldsymbol{X} \end{aligned} \qquad \textbf{(2.1)}$$

$$\boldsymbol{y} = (y_1, y_2, \ldots, y_k) \in \boldsymbol{Y}$$

siendo $\boldsymbol{x}$ el vector de decisión e $\boldsymbol{y}$ el vector objetivo. El espacio de decisión se denota por $\boldsymbol{X}$, y el espacio objetivo por $\boldsymbol{Y}$. Optimizar, dependiendo del problema, puede significar igualmente, minimizar o maximizar. El conjunto de restricciones $\boldsymbol{e(x)} \geq \boldsymbol{0}$ determina el conjunto de soluciones factibles $\boldsymbol{X}_f \subset \boldsymbol{X}$ y su correspondiente conjunto de vectores objetivo factibles $\boldsymbol{Y}_f \subset \boldsymbol{Y}$.

Definición 2: Conjunto de soluciones factibles. El conjunto de soluciones factibles $\boldsymbol{X_f}$ se define como el conjunto de vectores de decisión $\boldsymbol{x}$ que satisface $\boldsymbol{e}(\boldsymbol{x})$:

$$\boldsymbol{X_f} = \{\boldsymbol{x} \in \boldsymbol{X} \mid \boldsymbol{e}(\boldsymbol{x}) \geq \boldsymbol{0}\} \qquad \textbf{(2.2)}$$

La imagen de $\boldsymbol{X_f}$, es decir, la región factible del espacio objetivo, se denota por

$$\boldsymbol{Y_f} = \boldsymbol{f}(\boldsymbol{X_f}) = \bigcup_{x \in X_f} \{\boldsymbol{y} = \boldsymbol{f}(\boldsymbol{x})\} \qquad \textbf{(2.3)}$$

De estas definiciones se tiene que cada solución del MOP en cuestión consiste de una n-tupla $\boldsymbol{x}=(x_1, x_2, \ldots, x_n)$, que conduce a un vector objetivo $\boldsymbol{y} = (f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), \ldots, f_k(\boldsymbol{x}))$, donde cada $\boldsymbol{x}$ debe cumplir con el conjunto de restricciones $\boldsymbol{e}(\boldsymbol{x}) \geq \boldsymbol{0}$. El problema de optimización consiste en hallar la $\boldsymbol{x}$ que tenga el "mejor valor" de $\boldsymbol{f}(\boldsymbol{x})$. En general, no existe un único "mejor valor", sino un conjunto de soluciones. Entre éstas, ninguna se puede considerar mejor que las demás si se tienen en cuenta todos los objetivos simultáneamente.

Definición 3: Extensión de las relaciones =, $\leq$ y $\geq$ a MOPs.

Dados 2 vectores de decisión $\boldsymbol{u}, \boldsymbol{v} \in \boldsymbol{X}$, en un contexto de minimización se define,

$$\begin{aligned}
&\boldsymbol{f}(\boldsymbol{u}) = \boldsymbol{f}(\boldsymbol{v}) \quad \text{si y solo si} \quad \forall i \in \{1, 2, \ldots, k\}: f_i(\boldsymbol{u}) = f_i(\boldsymbol{v}) \\
&\boldsymbol{f}(\boldsymbol{u}) \leq \boldsymbol{f}(\boldsymbol{v}) \quad \text{si y solo si} \quad \forall i \in \{1, 2, \ldots, k\}: f_i(\boldsymbol{u}) \leq f_i(\boldsymbol{v}) \qquad \textbf{(2.4)} \\
&\boldsymbol{f}(\boldsymbol{u}) < \boldsymbol{f}(\boldsymbol{v}) \quad \text{si y solo si} \quad \boldsymbol{f}(\boldsymbol{u}) \leq \boldsymbol{f}(\boldsymbol{v}) \wedge \boldsymbol{f}(\boldsymbol{u}) \neq \boldsymbol{f}(\boldsymbol{v})
\end{aligned}$$

Las relaciones $\geq$ y $>$ se definen de manera similar en un contexto de maximización. Es claro que dos vectores $\boldsymbol{u}, \boldsymbol{v} \in \boldsymbol{X}$ solo pueden cumplir con una de tres condiciones que se expresan con los siguientes símbolos y términos:

Definición 4: Dominancia Pareto en un contexto de minimización.

Para dos vectores de decisión $\boldsymbol{u}$ y $\boldsymbol{v}$,

$\boldsymbol{u} \succ \boldsymbol{v}$ ($\boldsymbol{u}$ domina a $\boldsymbol{v}$)	si y solo si	$\boldsymbol{f}(\boldsymbol{u}) < \boldsymbol{f}(\boldsymbol{v})$	
$\boldsymbol{v} \succ \boldsymbol{u}$ ($\boldsymbol{v}$ domina a $\boldsymbol{u}$)	si y solo si	$\boldsymbol{f}(\boldsymbol{v}) < \boldsymbol{f}(\boldsymbol{u})$	**(2.5)**
$\boldsymbol{u} \sim \boldsymbol{v}$ ($\boldsymbol{u}$ y $\boldsymbol{v}$ no son comparables)	si y solo si	$\boldsymbol{f}(\boldsymbol{u}) \nsucc \boldsymbol{f}(\boldsymbol{v}) \wedge \boldsymbol{f}(\boldsymbol{v}) \nsucc \boldsymbol{f}(\boldsymbol{u})$	

Definición 5: Dominancia Pareto en un contexto de maximización.
Para dos vectores de decisión $\boldsymbol{u}$ y $\boldsymbol{v}$,

$\boldsymbol{u} \succ \boldsymbol{v}$ ($\boldsymbol{u}$ domina a $\boldsymbol{v}$)	si y solo si	$\boldsymbol{f}(\boldsymbol{u}) > \boldsymbol{f}(\boldsymbol{v})$	
$\boldsymbol{v} \succ \boldsymbol{u}$ ($\boldsymbol{v}$ domina a $\boldsymbol{u}$)	si y solo si	$\boldsymbol{f}(\boldsymbol{v}) > \boldsymbol{f}(\boldsymbol{u})$	**(2.6)**
$\boldsymbol{u} \sim \boldsymbol{v}$ ($\boldsymbol{u}$ y $\boldsymbol{v}$ no son comparables)	si y solo si	$\boldsymbol{f}(\boldsymbol{u}) \nsucc \boldsymbol{f}(\boldsymbol{v}) \wedge \boldsymbol{f}(\boldsymbol{v}) \nsucc \boldsymbol{f}(\boldsymbol{u})$	

Definición 6: Optimalidad Pareto. Dado un vector de decisión $\boldsymbol{x} \in \boldsymbol{X}_f$, se dice que $\boldsymbol{x}$ es no dominado respecto a un conjunto $\boldsymbol{V} \subseteq \boldsymbol{X}_f$ si y solo si

$$\forall\ \boldsymbol{v} \in \boldsymbol{V} : (\boldsymbol{x} \succ \boldsymbol{v} \vee \boldsymbol{x} \sim \boldsymbol{v}) \qquad \textbf{(2.7)}$$

En caso que $\boldsymbol{x}$ sea no dominado respecto a todo el conjunto $\boldsymbol{X}_f$, y solo en ese caso, se dice que $\boldsymbol{x}$ es una **solución Pareto óptima**. Por lo tanto, el conjunto Pareto óptimo $\boldsymbol{X}_{true}$ puede ser definido formalmente a continuación.

Definición 7: Conjunto Pareto óptimo. Dado el conjunto de vectores de decisión factibles $\boldsymbol{X}_f$, se denomina conjunto Pareto óptimo $\boldsymbol{X}_{true}$ al conjunto de vectores de decisión no dominados de $\boldsymbol{X}_f$, es decir:

$$\boldsymbol{X}_{true} = \{\, \boldsymbol{x} \in \boldsymbol{X}_f \mid \boldsymbol{x} \text{ es no-dominado con respecto a } \boldsymbol{X}_f \,\} \qquad \textbf{(2.8)}$$

El correspondiente conjunto de vectores objetivos $\boldsymbol{Y}_{true} = \boldsymbol{f}(\boldsymbol{X}_{true})$ constituye el **frente Pareto óptimo** [CRI04]**.**

2.3. Problemas ZDT

En [DEB98] se identifican diferentes características que pueden hacer que los MOEAs tengan dos tipos de dificultades:

1. Converger al conjunto Pareto óptimo. La multimodalidad, la decepción y los óptimos aislados son áreas problemáticas bien conocidas que dificultan la convergencia de los EAs [DEB99]. Los problemas multimodales son los que tienen múltiples óptimos locales o múltiples óptimos globales (múltiples soluciones al problema). En la resolución del problema se desea obtener varios de esos óptimos globales.
2. Mantener la diversidad en la población, la cual es necesaria para la obtención de un frente Pareto bien distribuido. De acuerdo al problema considerado, ciertas características propias del conjunto Pareto óptimo correspondiente pueden dificultar que el MOEA encuentre una variedad de soluciones Pareto-óptimas bien distribuidas:
   - convexidad o no-convexidad
   - discretitud
   - no-uniformidad

En [ZIT99] se desarrolló un conjunto de seis problemas de prueba que contemplan las distintas posibilidades existentes en [DEB98]. Todos son del tipo minimización y por tanto no se consideran los de tipo maximización y los que mezclan maximización/minimización.

Estos problemas son llamados ZDT en honor a los que los propusieron: Eckart **Z**itzler, Kalyanmoy **D**eb y Lothar **T**hiele, cabe resaltar que estas tres personas participaron en la elaboración de seis de los siete MOEAs utilizados en este trabajo.

Las funciones objetivos del conjunto de prueba se restringen a dos objetivos por considerarse la manera más simple y completa de reflejar los aspectos esenciales de la optimización multiobjetivo, cada una de ellas está estructurada de igual manera sobre tres funciones f_1 , g y h :

$$
\begin{aligned}
&\textit{Minimizar} && \boldsymbol{F}(\boldsymbol{x}) = (f_1(x_1), f_2(\boldsymbol{x})) \\
&\textit{sujeta a} && f_2(\boldsymbol{x}) = \boldsymbol{g}(x_2, \ldots, x_n) \cdot \boldsymbol{h}\,(f_1(x_1), \boldsymbol{g}(x_2, \ldots, x_n)) \qquad \textbf{(2.9)} \\
&\textit{donde} && \boldsymbol{x} = (x_1, x_2, \ldots, x_n) \in \boldsymbol{X}
\end{aligned}
$$

La función f_1 es una función que depende únicamente de la primera variable de decisión, g es una función de las n-1 variables de decisión restantes y los parámetros de h son los valores de las funciones f_1 y g. Las funciones de prueba difieren en estas tres funciones así como en el número de variables n y en los valores que éstas pueden tomar.

2.3.1. Problema ZDT1

El problema ZDT1 tiene 30 variables en el rango [0,1]. Su frente de Pareto es convexo. Es el problema más sencillo del conjunto, tiene un frente de Pareto continuo y una distribución uniforme de soluciones a lo largo del frente.

$$f_1(x_1) = x_1$$

$$g(x_2,\ldots,x_n) = 1 + 9 \cdot \sum_{i=2}^{n} \frac{x_i}{(n-1)} \qquad \textbf{(2.10)}$$

$$h(f_1, g) = 1 - \sqrt{\frac{f_1}{g}}$$

donde: $x = (x_1,\ldots,x_n)$, n = 30 y $x_i \in [0,1]$.

El frente Pareto-óptimo se forma con $g(x_2,\ldots,x_n) = 1$ como se puede apreciar en la Figura 4.

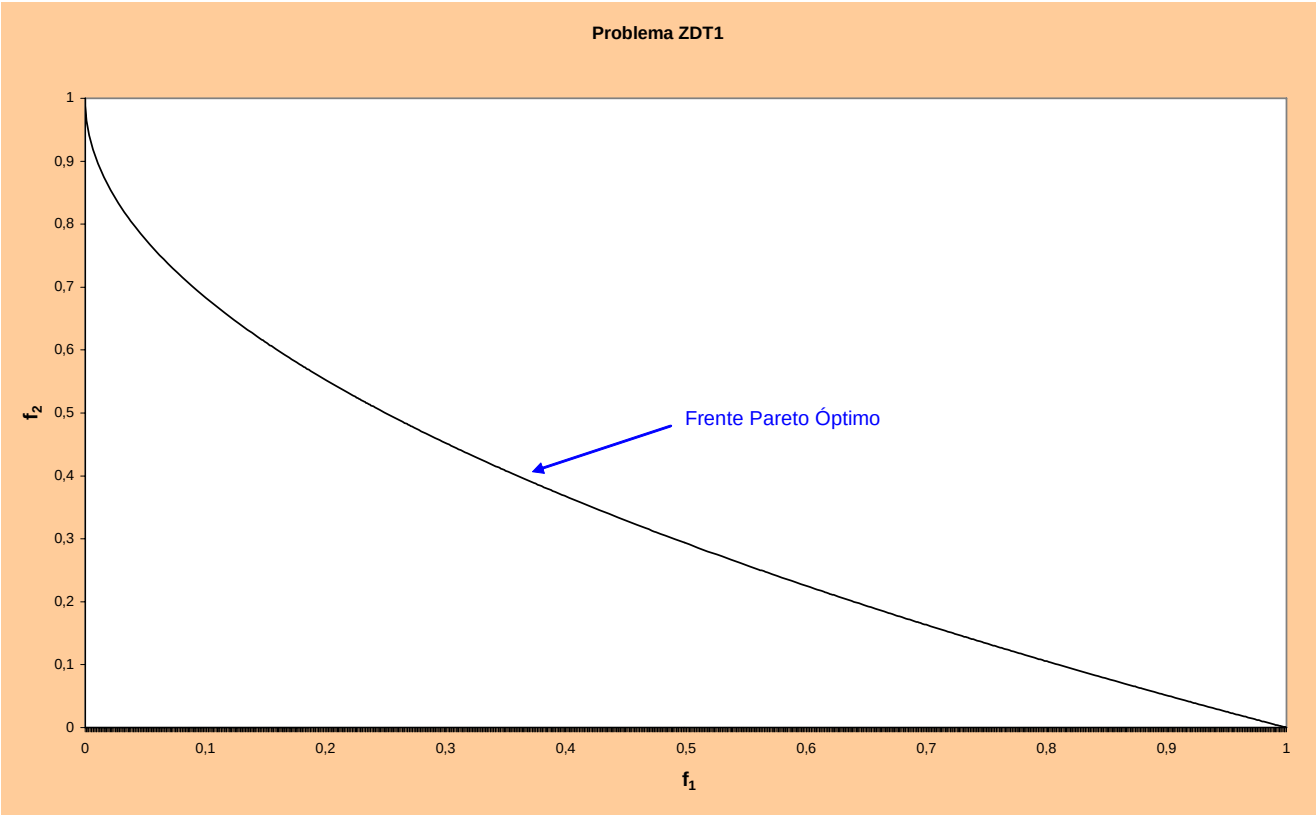

Figura 4. Frente Pareto Óptimo para el problema ZDT1

2.3.2. Problema ZDT2

El problema ZDT2 tiene 30 variables en el rango [0,1]. Su frente de Pareto es no convexo. La distribución de soluciones a lo largo del frente de Pareto es uniforme.

$$f_1(x_1) = x_1$$

$$g(x_2,\ldots,x_n) = 1 + 9 \cdot \sum_{i=2}^{n} \frac{x_i}{(n-1)} \qquad \textbf{(2.11)}$$

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

donde: $x = (x_1,\ldots,x_n)$, n = 30 y $x_i \in [0,1]$.

En la figura 5 se aprecia el frente Pareto-óptimo, el cual se obtiene con $g(x_2,\ldots,x_n) = 1$.

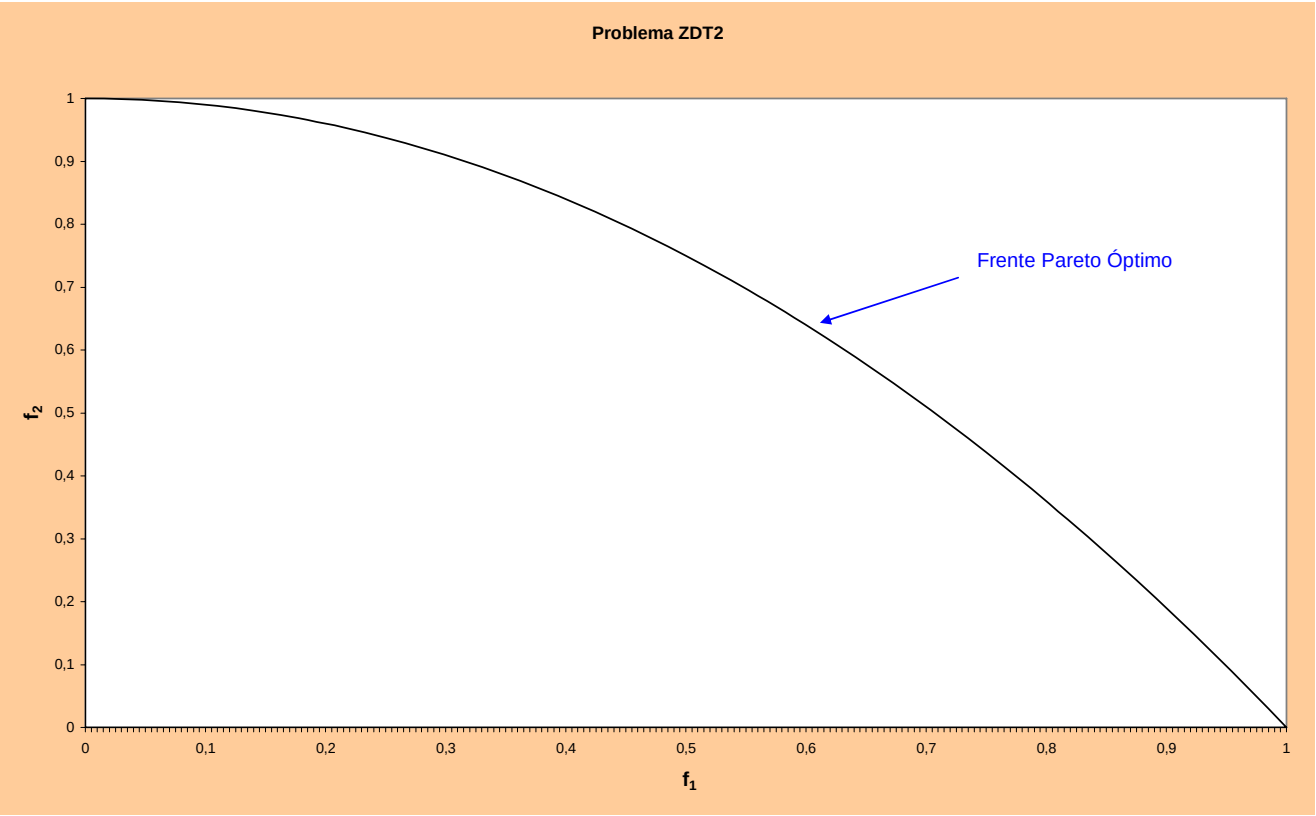

Figura 5. Frente Pareto Óptimo para el problema ZDT2

2.3.3. Problema ZDT3

El problema ZDT3 tiene 30 variables en el rango [0,1]. Su frente de Pareto es discontinuo. La distribución de soluciones a lo largo del frente es uniforme.

$$f_1(x_1) = x_1$$

$$g(x_2,\ldots,x_n) = 1 + 9 \cdot \sum_{i=2}^{n} \frac{x_i}{(n-1)} \qquad \textbf{(2.12)}$$

$$h(f_1, g) = 1 - \sqrt{\frac{f_1}{g}} - \left(\frac{f_1}{g}\right) sin(10\pi f_1)$$

donde: $x = (x_1,\ldots,x_n)$, n = 30 y $x_i \in [0,1]$.

La utilización de la función seno en ***h*** causa la discontinuidad en el frente Pareto-óptimo. Sin embargo, no existe discontinuidad en el espacio de parámetros. El frente Pareto-óptimo se forma con $g(x_2,\ldots,x_n)=1$ y se observa en la figura 6, donde también se aprecian las discontinuidades.

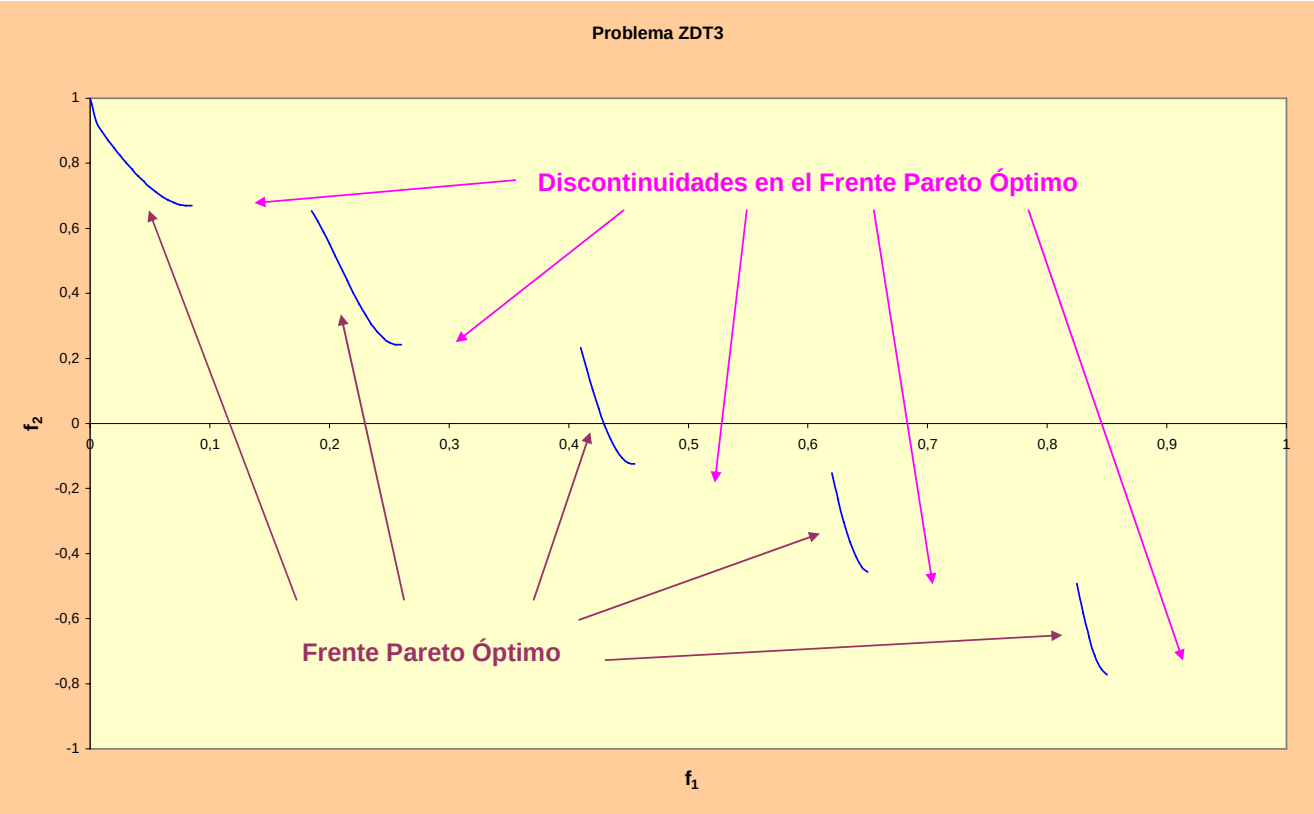

Figura 6. Frente Pareto Óptimo para el problema ZDT3

2.3.4. Problema ZDT4

El problema ZDT4 tiene 10 variables en el rango [0,1]. Su frente de Pareto es convexo. La complejidad de este problema es que contiene 21^9 frentes Pareto-óptimos locales [ZIT99] y, por tanto, prueba a los algoritmos evolutivos con relación a su capacidad de lidiar con la multimodalidad:

$$f_1(x_1)=x_1$$

$$g(x_2,\ldots,x_n)=1+10(n-1)+\sum_{i=2}^{n}\left(x_i^2-10\cos(4\pi x_i)\right) \qquad \textbf{(2.13)}$$

$$h(f_1, g) = 1 - \sqrt{\frac{f_1}{g}}$$

donde: $x = (x_1, \ldots, x_n)$, n = 10 , $x_1 \in [0,1]$, $x_2 \ldots x_n \in [-5,5]$.

El conjunto Pareto-óptimo global se obtiene al ser $g(x_2, \ldots, x_n) = 1$ y el mejor frente Pareto-óptimo local con $g(x_2, \ldots, x_n) = 1{,}25$ como se aprecian en la Figura 7. Note que no todos los conjuntos Pareto-óptimos locales son distinguibles en el espacio objetivo.

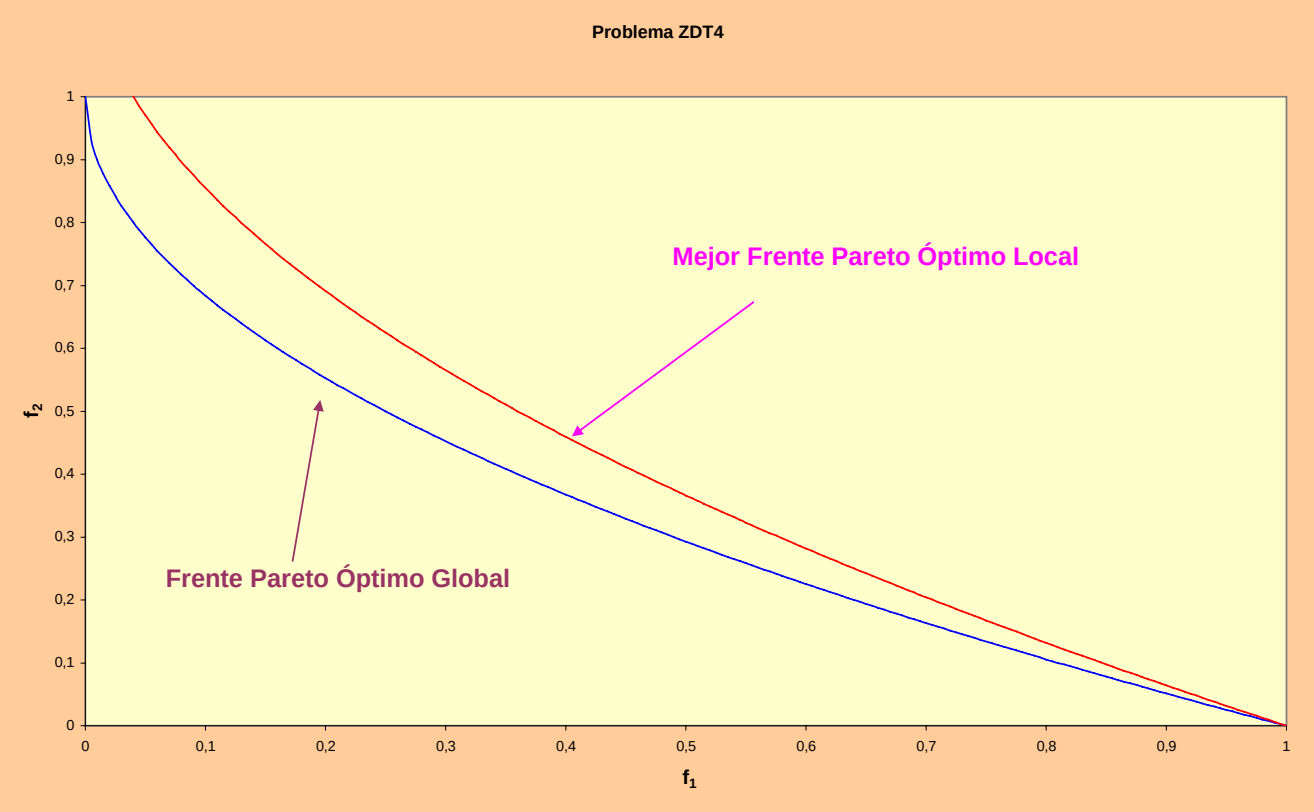

Figura 7. Frente Pareto Óptimo para el problema ZDT4

2.3.5. Problema ZDT5

El problema ZDT5 utiliza funciones booleanas definidas sobre strings. Constituye un caso de problema *deceptivo* ya que la forma de la función auxiliar determina que la mayor parte del espacio de

búsqueda se concentra cerca de óptimos locales, mientras que el óptimo global se halla relativamente aislado.

$$f_1(x_1) = 1 + u(x_1)$$

$$g(x_2,\ldots,x_n) = \sum_{i=2}^{n} v(u(x_i))$$

$$h(f_1, g) = \frac{1}{f_1} \qquad \textbf{(2.14)}$$

donde: $x = (x_1,\ldots,x_n)$, n = 11 , $x_1 \in \{0,1\}^{30}$, $x_2 \ldots x_n \in \{0,1\}^{5}$

$$u(x_i) = \text{"Número de unos en } x_i\text{"}, \quad v(u(x_i)) = \begin{cases} 2 + u(x_i) & \text{si } u(x_i) < 5 \\ 1 & \text{si } u(x_i) = 5 \end{cases}$$

El conjunto Pareto-óptimo global se forma con $g(x_2,\ldots,x_n)$=10 y el mejor frente Pareto-óptimo local con $g(x_2,\ldots,x_n)$= 11 como se visualiza en la Figura 8.

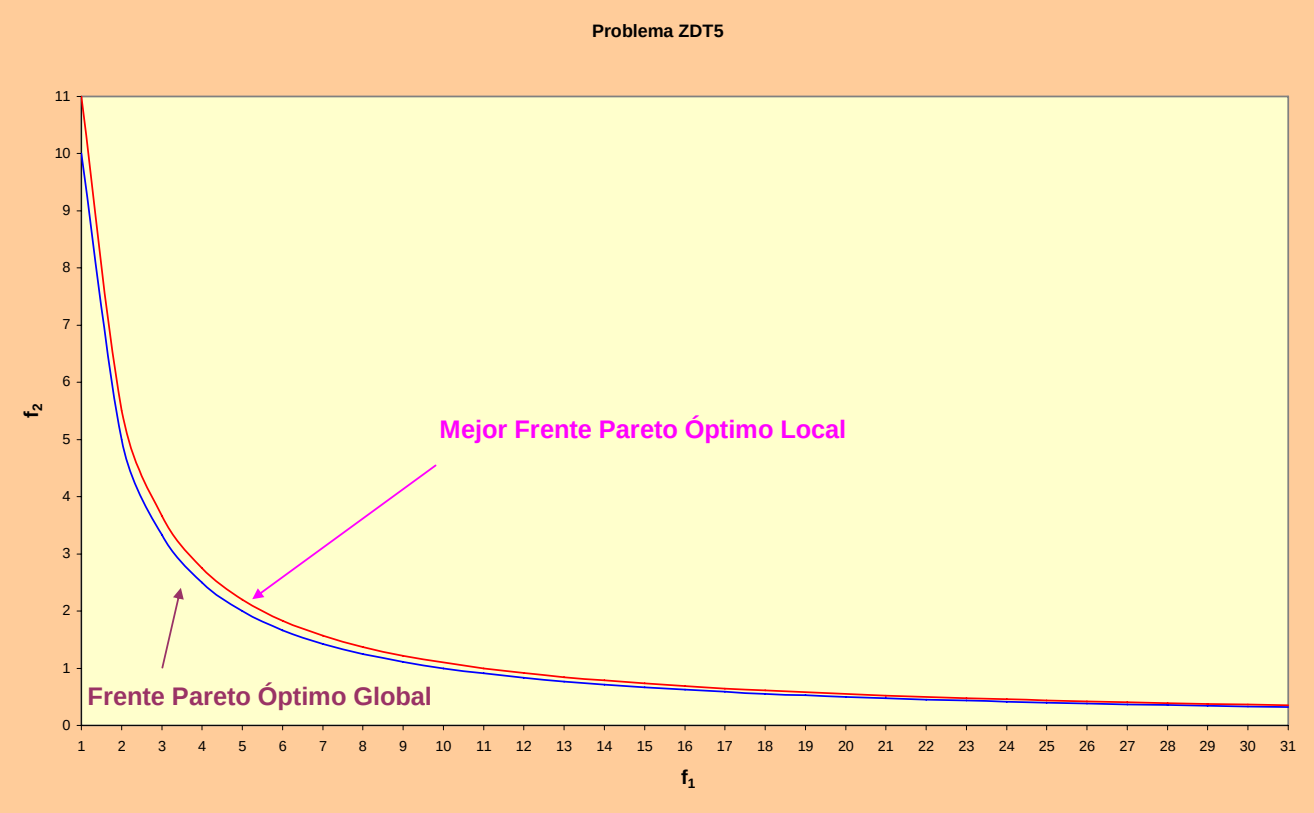

Figura 8. Frente Pareto Óptimo para el problema ZDT5

2.3.6. Problema ZDT6

El problema ZDT6 tiene 10 variables en el rango [0,1]. Su frente de Pareto es no convexo. La complejidad de este problema está dada por la combinación de la forma no convexa del frente Pareto y la distribución no uniforme de soluciones a lo largo de él.

$$
\begin{aligned}
&f_1(x_1) = 1 - e^{(-4x_1)}\,\mathbf{sin}^6(6\pi x_1)\\
&g(x_2,\ldots,x_n) = 1 + 9\cdot\left(\sum_{i=2}^{n}\frac{x_i}{(n-1)}\right)^{\frac{1}{4}}\\
&h(f_1,g) = 1 - \left(\frac{f_1}{g}\right)^2
\end{aligned}
\qquad \textbf{(2.15)}
$$

donde: $x = (x_1,\ldots,x_n)$, n = 10 y $x_i \in [0,1]$.

El frente Pareto-óptimo se consigue al ser $g(x_2,\ldots,x_n) = 1$ como se representa en la Figura 9.

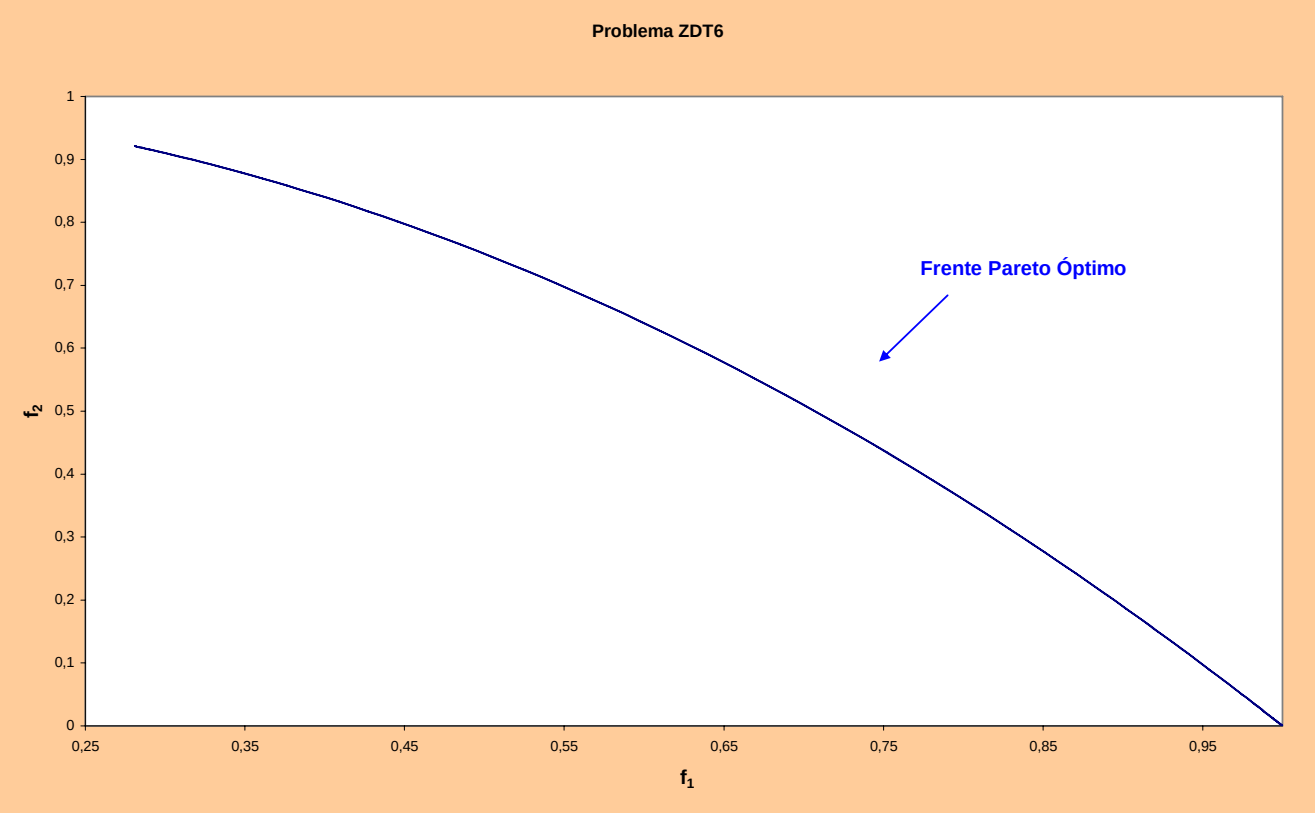

Figura 9. Frente Pareto Óptimo para el problema ZDT6

2.4. Proceso de solución de un MOP

Un problema de optimización multiobjetivo se considera matemáticamente resuelto cuando el conjunto Pareto Óptimo es encontrado. En problemas reales, usualmente se requiere de una única solución.

Puesto que desde un punto de vista estrictamente matemático todas las soluciones en el conjunto Pareto son igualmente buenas, para obtener la solución a ser implementada en un caso concreto, se necesita información adicional de preferencia. Un tomador de decisiones (*decision maker* - DM) humano es quien se encarga de proporcionar la información de preferencia necesaria para seleccionar una solución del conjunto Pareto. Así, el proceso de solución de un MOP puede dividirse en dos procesos conceptualmente distintos [HOR93, MIE01]:

- **Proceso de búsqueda u optimización**, por el cual se explora el conjunto de soluciones factibles en busca de soluciones Pareto Optimas.
- **Proceso de toma de decisiones**, por el cual se selecciona una solución de compromiso adecuada, a partir del Conjunto Pareto Óptimo hallado por el proceso anterior.

La importancia de la toma de decisiones como parte del proceso de solución es una buena razón para clasificar los distintos métodos para solucionar problemas de optimización multiobjetivo de acuerdo con la manera en que se combinan la búsqueda del conjunto de soluciones y la toma de decisiones [MIE01]. Así, los métodos se clasifican en:

- **Métodos a priori:** la toma de decisiones se realiza antes de la búsqueda de soluciones. Usualmente los distintos objetivos se combinan en uno, el cual implícitamente incluye información de preferencia proporcionada por el tomador de decisiones. Esto hace que efectivamente el MOP se convierta en un problema de optimización monobjetivo, antes de la optimización.
- **Métodos a posteriori:** la toma de decisiones se realiza de forma posterior a la búsqueda. La optimización se realiza sin ninguna información de preferencia. El resultado del proceso de búsqueda es un conjunto de soluciones candidatas, idealmente Pareto-Optimas, a partir de la cual la decisión final será realizada por el tomador de decisiones, conociendo las alternativas disponibles.
- **Métodos interactivos o progresivos:** la toma de decisiones se realiza durante la búsqueda de soluciones de manera interactiva. Luego de cada paso de optimización, un conjunto de soluciones de compromiso es presentado al tomador de decisiones, quien proporcionará información de preferencia que guiará el proceso de búsqueda.

Con los métodos a priori, donde se combinan múltiples objetivos en un único criterio de optimización, cuando son aplicables, se tiene la ventaja que las estrategias clásicas de optimización monobjetivo pueden ser utilizadas sin modificaciones. La desventaja es que requiere un conocimiento profundo del dominio del problema que permita realizar la escalarización de forma correcta. Este conocimiento del dominio del problema requerido por los métodos a priori usualmente no se encuentra disponible. Es más, en varios problemas de diseño en ingeniería, lo que se desea específicamente es ganar mayor conocimiento sobre el problema y sobre las soluciones alternativas. Realizar la toma de decisiones luego del proceso de búsqueda (métodos a posteriori) resuelve el problema, pero excluye la

articulación de la preferencia del tomador de decisiones, lo que podría a su vez reducir la complejidad del espacio de búsqueda. Los métodos interactivos superan las desventajas de los métodos a priori y posteriori permitiendo al tomador de decisiones expresar sus preferencias a medida que avanza en el conocimiento del problema considerado.

2.5. Dificultades de los métodos tradicionales

Numerosos métodos clásicos existen para el tratamiento de problemas de optimización multiobjetivo. Todos ellos se caracterizan por operar en dos fases. Primeramente generan, a partir del problema de objetivos múltiples, un problema de optimización de un solo objetivo (*Single Objective Problem* - SOP). En la segunda fase aplican un método de optimización tradicional al SOP generado en la fase anterior y obtienen sus resultados. Ambas fases son independientes entre sí. Una vez obtenido el SOP se puede resolverlo con cualquier técnica típica (*hill climbing*, algoritmos de búsqueda aleatoria, métodos numéricos, etc.).

Varios tipos de métodos se emplean para la reducción realizada en la primera fase, cada uno de ellos tiene sus propias ventajas y desventajas. En gran parte de ellos, la técnica consiste en disponer de las funciones objetivo de modo tal que se puedan unir en una única función parametrizada, mientras el proceso de optimización se aplica sobre esta única función. Generalmente, los parámetros de la función se varían de modo sistemático y se requieren múltiples corridas del procedimiento de optimización (2da. fase) para conseguir un conjunto que se aproxime al conjunto Pareto óptimo real.

Algunas técnicas representativas son “Método de la suma con pesos” [COH78], “Método de las restricciones” [COH78], “Programación en base a objetivos [STE86], y la “Aproximación Min Max” [STE86]. Ejemplos de métodos empleados en la segunda fase son: búsqueda exhaustiva, programación lineal, optimización utilizando el concepto del gradiente, técnicas de inteligencia artificial, etc.

Los distintos métodos tradicionales cuentan con al menos una de las siguientes dificultades [DEB99]:

1. El algoritmo de optimización se debe aplicar varias veces para encontrar un conjunto de soluciones Pareto óptimas. Como cada corrida es independiente de las demás,

generalmente no se obtiene un efecto sinergético. Por tanto, delinear el frente Pareto óptimo resulta costoso en términos computacionales.

2. La mayoría de los algoritmos requieren conocimiento previo del problema a resolver y son muy sensibles a los parámetros del algoritmo: pesos de los objetivos, orden de evaluación, nivel de objetivos, valor de las restricciones, etc.
3. Algunos algoritmos son sensibles a la forma del frente Pareto.
4. La variación entre las diferentes soluciones encontradas depende de la eficiencia del optimizador de un sólo objetivo. Podría darse el caso de encontrar siempre la misma solución o soluciones muy parecidas en distintas corridas.
5. No son necesariamente confiables en problemas que involucren al azar o incertezas.

Investigaciones recientes han demostrado que las dificultades arriba mencionadas pueden ser superadas con la utilización de algoritmos evolutivos [DEB99]. En el siguiente capítulo se presentan estos algoritmos aplicados a optimizaciones con objetivos múltiples.

Capítulo 3

Algoritmos Evolutivos Multiobjetivo

Los MOEAs (Algoritmos Evolutivos para la Optimización Multiobjetivo) son herramientas algorítmicas desarrolladas recientemente para la resolución de MOPs.

Su popularidad se debe principalmente a que:

i. resultan promisorios para la integración de los aspectos de búsqueda y de toma de decisiones que plantean los MOPs;
ii. pueden realizar búsquedas aún en espacios ilimitados y altamente complejos;
iii. tienen la habilidad de mantener toda una población de soluciones;
iv. Son una alternativa válida para el tratamiento adecuado de MOPs.

Este capítulo presenta los conceptos de los algoritmos evolutivos, el proceso de un algoritmo evolutivo en su forma genérica. Luego se detallan los siete MOEAs utilizados en el presente trabajo.

Y para terminar en la sección 3.4 se introducen los MOEAs paralelos, sus modelos más utilizados y las ventajas que poseen.

3.1. Algoritmos Evolutivos

El proceso de evolución puede ser visto como una búsqueda entre el conjunto enorme de posibles secuencias de genes de aquellas que correspondan a los mejores individuos. Esta búsqueda que la naturaleza realiza es robusta, en el sentido que es capaz de localizar eficaz y eficientemente individuos de mejor adaptación en un espacio de búsqueda enorme y cambiante.

Las reglas de la evolución son notablemente simples. Las especies evolucionan por medio de la selección natural, esto es, la conservación de las diferencias y variaciones individualmente favorables y la destrucción de las que son perjudiciales. Los individuos que tienen ventaja sobre otros, por ligera que sea, tendrán más probabilidades de sobrevivir y procrear su especie [DAR92].

Los hijos son semejantes a sus padres, por lo que cada nueva generación tiene individuos semejantes a los mejores individuos de la generación anterior. Además del proceso de selección natural, en la naturaleza ocasionalmente se producen mutaciones al azar. Algunas de estas mutaciones conducirán a nuevos y mejores individuos, de esta manera, alteraciones aleatorias serán introducidas y luego propagadas a generaciones futuras. Por el contrario, las características negativas introducidas por la mutación tenderán a desaparecer pues conducen a individuos pobremente adaptados. Si las modificaciones del ambiente son menores, las especies irán evolucionando gradualmente con éste; sin embargo, es probable que un súbito cambio de ambiente provoque la desaparición de especies enteras y el nacimiento de otras nuevas, capaces de sobrevivir en el nuevo entorno.

Los algoritmos evolutivos son algoritmos de búsqueda y optimización que simulando los mecanismos esenciales de la evolución Darwiniana, intentan reproducir las características de robustez y simplicidad existentes en la naturaleza para evolucionar hacia mejores soluciones en problemas computacionales [GOL89].

Estos algoritmos trabajan con un conjunto de posibles soluciones para el problema que se intenta resolver. Cada una de estas soluciones representa un individuo y todo el conjunto representa una población. La información relativa a las características de los individuos en poblaciones naturales viene dada en los cromosomas, los cuales son responsables de la carga hereditaria. Por ello, cuando se trabaja con EAs, las soluciones del problema considerado se codifican en forma de cadenas semejantes a las cadenas de genes que conforman los cromosomas.

En la naturaleza, cada individuo posee unas características particulares, que miden de alguna manera su capacidad de adaptabilidad con respecto a otros individuos de la misma especie en un entorno dado. Del mismo modo, en los EAs, a cada solución de la población artificial se le asigna un valor de adaptabilidad (*fitness*) de acuerdo a que tan buena sea como solución del problema en cuestión. Dicho valor servirá para que un proceso aleatorio de selección artificial, que favorece a los individuos con mejor adaptabilidad, tome pares de individuos de la población para ser sometidos a un operador de cruzamiento. El operador de cruzamiento implementa alguna forma de intercambio entre las cadenas que conforman los individuos seleccionados (padres), para generar nuevas soluciones (hijos) las cuales formarán una nueva población.

Los procedimientos de asignación de *fitness*, selección y cruzamiento se repiten de forma iterativa hasta que algún criterio de parada se cumpla. A cada iteración se le denomina comúnmente generación.

Entonces, se dice que un nuevo conjunto de soluciones se crea en cada generación por medio de la selección de buenos individuos de la generación actual y el cruzamiento de los mismos.

Los miembros de cada nueva generación creada a partir de la selección y el cruzamiento tendrán características similares a las de los mejores individuos (soluciones) de la generación anterior. Sin embargo, ciertas características importantes pueden perderse en este proceso. Un operador de mutación que altera en forma aleatoria una solución tiene a su cargo recuperar la información que se pueda perder durante el proceso simulado de evolución natural, introduciendo una búsqueda aleatoria en el espacio de soluciones. Este proceso análogo al llevado a cabo en entornos naturales, proporciona a los EAs las características necesarias para un comportamiento robusto que le permite ser utilizado para un amplio rango de problemas de optimización de funciones, optimización combinatoria, sistemas de aprendizajes de lógica difusa, aprendizaje de redes neuronales, entre otros [CLE97, DEB99, VEL99, ZIT01].

3.1.1. Representación de soluciones

Los algoritmos genéticos no trabajan directamente sobre las soluciones del problema en cuestión, sino que lo hacen sobre una abstracción de los objetos solución, usualmente denominadas cromosomas por analogía con la evolución natural biológica. Un cromosoma es un vector de genes, mientras que el valor asignado a un gen se denomina alelo.

En la terminología biológica, genotipo denota al conjunto de cromosomas que definen las características de un individuo. El genotipo sometido al medio ambiente se denomina fenotipo. En términos de los algoritmos genéticos el genotipo también esta constituido por cromosomas, utilizándose generalmente un único cromosoma por individuo solución. Por ello suelen utilizarse indistintamente los términos genotipo, cromosoma e individuo. Por su parte, el fenotipo representa un punto del espacio de soluciones del problema.

Dado que un algoritmo genético trabaja sobre cromosomas, se debe definir una función de codificación sobre los puntos del espacio de soluciones, que mapea todo punto del espacio de soluciones en un genotipo, tal como se indica en la Figura 10. La función inversa de la codificación, denominada decodificación permite obtener el fenotipo asociado a un cromosoma.

Codificación : Espacio de soluciones → cromosoma

Figura 10. Especificación de la función de codificación de un algoritmo genético

Tomando en cuenta la observación anterior, los mecanismos de codificación de individuos solución resultan importantes para el proceso de búsqueda de los algoritmos genéticos. Habitualmente los algoritmos genéticos utilizan codificaciones binarias de largo fijo. Los individuos se codifican por un conjunto de cardinalidad conocida de valores binarios (ceros y unos) conocido como *string* de bits o *bitstring*. Cada *bitstring* representa a una solución potencial del problema de acuerdo al mecanismo de codificación predefinido, en general dependiente del problema.

Otros esquemas de codificación han sido utilizados con menor frecuencia en los algoritmos genéticos. En particular las codificaciones basadas en números reales son útiles para representar soluciones cuando se resuelven problemas sobre espacios de cardinalidad no numerable, como en el caso de determinación de parámetros en problemas de control o entrenamiento de redes neuronales. Los esquemas basados en permutaciones de enteros son útiles para problemas de optimización combinatoria que involucran hallar ordenamientos óptimos, como los problemas de *scheduling* o el reconocido *Traveling Salesman Problem*.

Codificaciones dependientes de los problemas se han propuesto con frecuencia, como un mecanismo que permite incorporar conocimiento específico en la resolución de problemas complejos.

La Figura 11 resume gráficamente la relación entre el espacio de soluciones de un problema, la población de cromosomas con la cual trabaja el algoritmo genético y las funciones de codificación y decodificación.

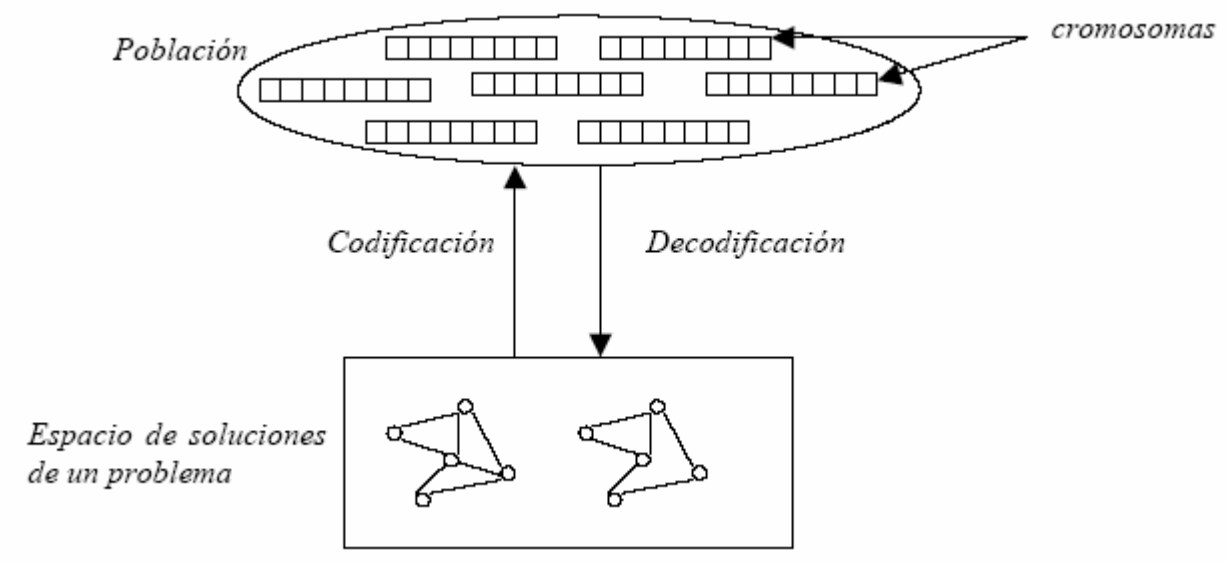

Figura 11. Codificación de soluciones en un algoritmo genético

3.1.2. Función de adaptación

Todo cromosoma tiene un valor asociado de *fitness* que evalúa la aptitud del individuo para resolver el problema en cuestión. La función de *fitness* tiene el mismo tipo que la función objetivo del problema, lo cual implica que el cálculo del valor de *fitness* se realiza sobre el fenotipo correspondiente al cromosoma, como se presenta en la Figura 12.

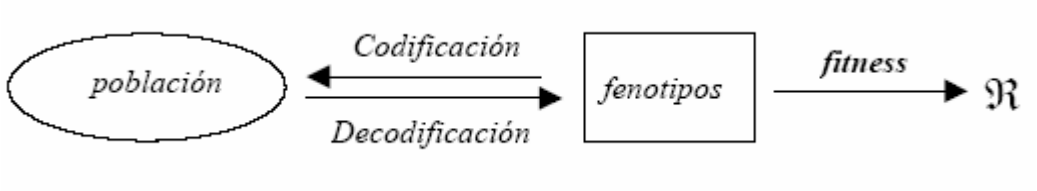

Figura 12. Codificación y función de *fitness*

La función de *fitness* tiene una influencia importante en el mecanismo del AG. Si bien actúa como una caja negra para el proceso evolutivo, la función de *fitness* guía el mecanismo de exploración, al actuar representando al entorno que evalúa la bondad de un individuo para la resolución del problema.

Varios puntos han sido identificados como importantes al momento de construir la función de *fitness*. De acuerdo a [COE02], es posible indicar que:

La función de *fitness* debe contemplar el criterio del problema de optimización (minimización o maximización de un objetivo) y las restricciones presentes en el problema de optimización. En caso de surgir soluciones no factibles del problema, la función de *fitness* deberá asignarle valores adecuados que garanticen que tales individuos no se perpetúen durante el proceso evolutivo (valores muy pequeños en el caso de un problema de maximización y muy elevados en el caso de un problema de minimización). A tales efectos diversas transformaciones a aplicar sobre funciones de *fitness* han sido definidas para mapear problemas de minimización a maximización y aplicar penalizaciones a soluciones no factibles [GOL89].

Deben contemplarse los casos en que el entorno presente problemas para la evaluación de la función de *fitness*, utilizando evaluaciones parciales cuando existen valores de *fitness* no definidos. Asimismo, debe considerarse el uso de funciones de *fitness* multivaluadas que asignen diferentes valores a un mismo individuo para simplificar la labor del operador de selección.

En general, la función de *fitness* será compleja de evaluar, en particular demandará un esfuerzo computacional mucho mayor que el requerido para los operadores evolutivos. Inclusive podría ocurrir que el proceso de evaluación sea tan complejo que solamente valores aproximados pudieran obtenerse en tiempos razonables. Este aspecto será importante al momento de proponer técnicas de alta performance para mejorar la eficiencia de los algoritmos genéticos.

En caso de problemas con objetivos múltiples, todos ellos deben estar contemplados en la función de *fitness*. La utilización de algoritmos genéticos para la resolución de problemas multiobjetivo constituye en sí misma una subárea de investigación con complejidades inherentes.

Para resolver problemas de dominancia de soluciones muy adaptadas en generaciones tempranas de la evolución, y para evitar el estancamiento en poblaciones similares al final de la evolución, deben considerarse mecanismos de escalado de los valores de *fitness* [MIC92].

3.1.3. Operadores Evolutivos

Los individuos en las poblaciones intercambian información por medio de operadores evolutivos. Existen tres operadores genéticos principales:

- **Selección:** Proceso por medio del cual los individuos son escogidos de la población de acuerdo al valor de su función de adaptación para someterse a la acción futura de otros operadores. Entre los métodos de selección más utilizados están la selección vía ruleta (*roulette-whell selection* - RW) y la selección por torneo (*tournament selection* - TS) [GOL89].
- **Cruzamiento:** Se inicia a partir de la obtención, utilizando algún operador de selección, de los individuos que serán sometidos al operador de cruzamiento que se encargará de intercambiar los componentes de los individuos seleccionados para producir nuevas soluciones. Así, el operador de cruzamiento se encarga de la transferencia por herencia de las características de los mejores individuos de una generación a la siguiente, siendo de esta manera la contraparte artificial de la recombinación sexual. Existen varias opciones para implementar el operador de cruzamiento siendo el cruzamiento de un sólo punto la forma más común de implementar este operador cuando se trabaja con una codificación de cadenas de bits. Este tipo de cruzamiento requiere la selección de dos cadenas como padres, luego, se selecciona un punto de cruzamiento, tal que dicho punto se encuentre entre 1 y (l-1), donde l es la longitud de la cadena. Los padres son cortados en este punto para producir cuatro subcadenas, que se intercambian para generar dos hijos. La Figura 13 muestra este procedimiento.

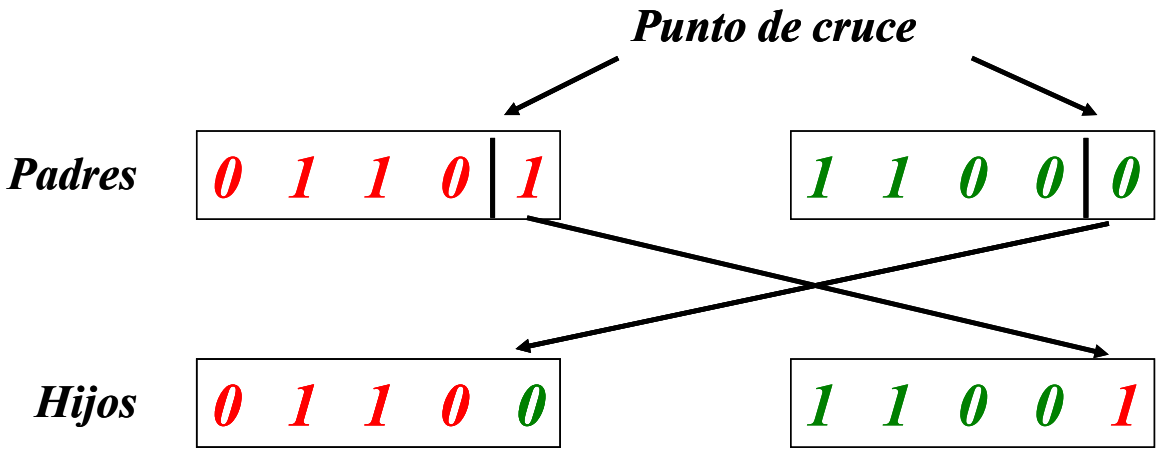

Figura 13. Funcionamiento del operador de un sólo punto

- **Mutación:** Utilizado como un operador de segundo plano cuyo propósito es la exploración aleatoria de nuevas porciones del espacio de búsqueda. Es este operador el encargado de introducir nuevo material genético en la búsqueda de soluciones ya que el cruzamiento no introduce ningún material nuevo. Cuando la codificación utilizada es binaria el operador de mutación simplemente modifica el valor de un bit en la cadena con una probabilidad de mutación generalmente pequeña.

3.1.4. Algoritmo Evolutivo Genérico

Una vez que se ha determinado la codificación, la función de adaptación, los parámetros y las variables para controlar el algoritmo y un criterio para terminar una corrida, un algoritmo evolutivo genérico puede ejecutarse implementando el algoritmo de la Figura 14.

Paso 1 - Inicializar la Población t=0
Paso 2 - Asignar *Fitness*
Paso 3 - Selección
Paso 4 - Cruzamiento
Paso 5 - Mutación
Paso 6 - Obtener Población Evolucionada t=t+1
Paso 7 - Sino se alcanzo un criterio de parada volver al Paso 2
Paso 8 – Parar

Figura 14 . Algoritmo Evolutivo Genérico

Todo el proceso de un algoritmo evolutivo aplicado a un problema de optimización multiobjetivo se resume en la Figura 15.

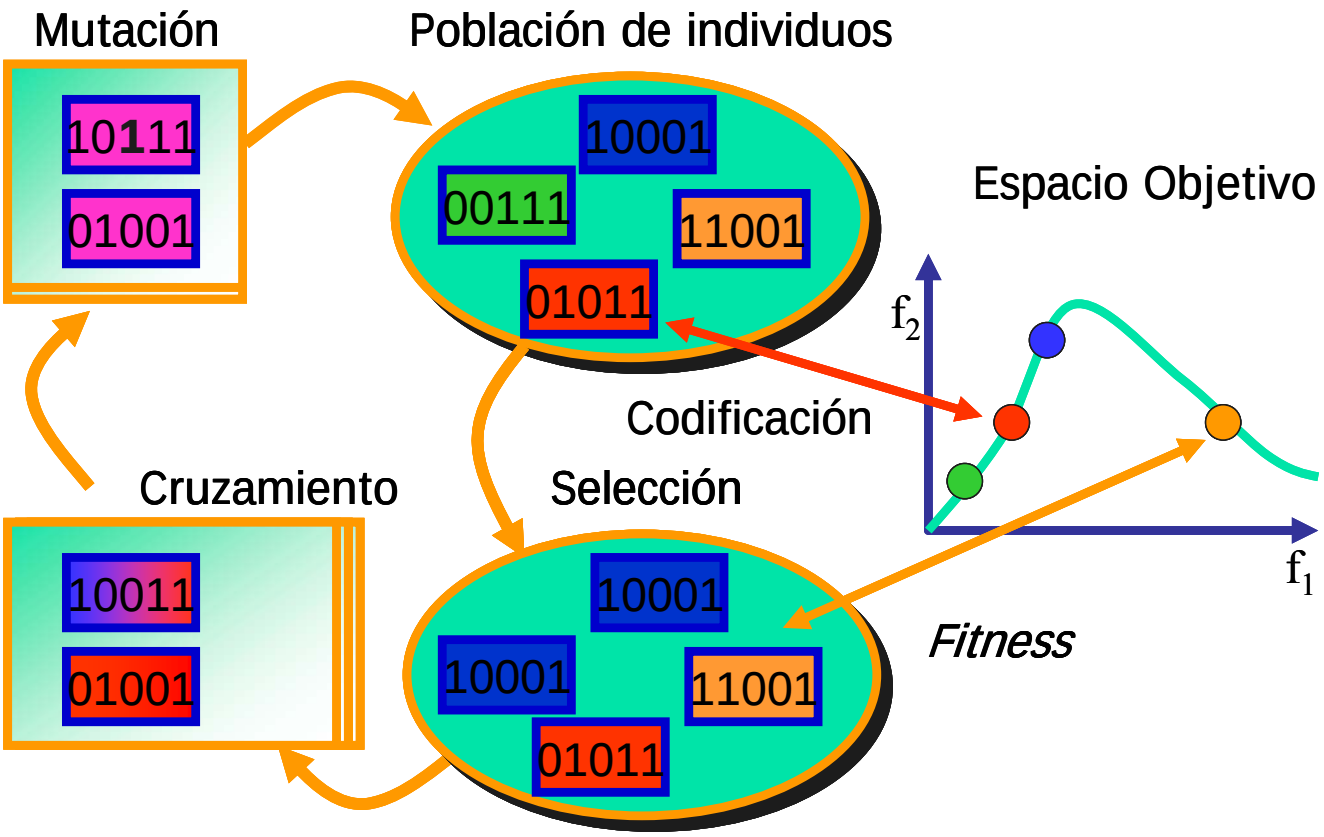

Figura 15. Proceso de un MOEA

3.2. MOEAs de primera generación

La falta de métodos determinísticos eficientes y eficaces para la resolución de problemas con objetivos múltiples, motivó la búsqueda de métodos alternativos. El notable éxito obtenido por los algoritmos evolutivos en optimización monobjetivo y las características propias de los mismos despertaron el interés de los investigadores en utilizarlos en optimización multiobjetivo. En la actualidad, la optimización evolutiva multiobjetivo (*Evolutionary Multiobjective Optimization* - EMOO) es un área de investigación muy importante tanto para científicos como para ingenieros, no sólo por que la mayor parte de los problemas consideren por naturaleza objetivos múltiples, sino también porque aún quedan por resolver un sin número de interrogantes en esta disciplina.

Como una muestra de lo incipiente del área, el primer congreso internacional de EMOO se realizó en marzo del 2001 [ZIT01]. En poco tiempo, la computación evolutiva multiobjetivo, se ha

establecido como el método para aproximar el frente Pareto-óptimo en problemas de este tipo. Esto se debe fundamentalmente al paralelismo intrínseco de los algoritmos evolutivos que les permite explorar similaridades entre las soluciones de forma eficiente y a la capacidad de capturar varias soluciones Pareto-óptimas en una única corrida de optimización [ZIT99].

Los diferentes métodos para trabajar con objetivos múltiples utilizando algoritmos evolutivos se clasifican en técnicas de primera y segunda generación. Pertenecen a la primera generación las propuestas iniciales que no consideran conceptos de Pareto. Así mismo, ésta generación abarca a los primeros algoritmos evolutivos multiobjetivo basados en Pareto que no incluyen mecanismos para la preservación de las buenas soluciones encontradas durante el proceso evolutivo (elitismo).

Un MOEA debe diseñarse para lograr dos propósitos en forma simultánea: lograr buenas aproximaciones al frente Pareto y mantener la diversidad de las soluciones, de modo a muestrear adecuadamente el espacio de soluciones y no converger a una solución única o a una sección acotada del frente. La Figura 16 presenta gráficamente los propósitos de un MOEA, donde se ha demarcado con la región celeste el espacio de búsqueda de funciones objetivo de un problema hipotético, mientras que la línea roja gruesa representa al frente de Pareto.

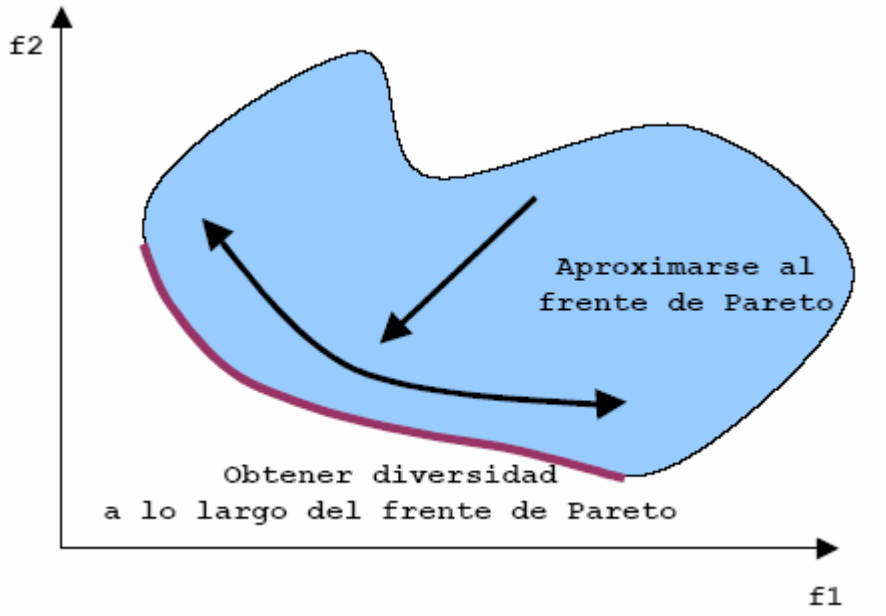

Figura 16. Propósitos de un MOEA

En la década de los noventa aparecen los primeros algoritmos que consideran de forma simultánea la optimización de objetivos múltiples utilizando el concepto de dominancia Pareto. El *Multiobjective Genetic Algorithm* (MOGA) de Fonseca y Flemming [FON93], el *Niched Pareto*

Genetic Algorithm (NPGA) de Horn y Nafpliotis [HOR93], y el *Nondominated Sorting Genetic Algorithm* (NSGA) de Srinivas y Deb [SRI93, SRI94] son de los primeros MOEAs basados en Pareto. Estos algoritmos, pertenecientes a la primera generación de algoritmos evolutivos multiobjetivo, fueron aplicados a una amplia gama de problemas multiobjetivo, donde demostraron un mejor desempeño que el de los enfoques no basados en Pareto y son los utilizados en el presente trabajo.

3.2.1. *Multiobjective Genetic Algorithm*

El MOGA (*Multiobjective Genetic Algorithm*) fue propuesto en [FON93] como un algoritmo genético multiobjetivo basado en una modificación al algoritmo genético simple en la etapa de selección. El algoritmo se basa en un método de clasificación (ranking) de los individuos de la población genética de acuerdo al número de elementos que lo dominan. A partir de la información del procedimiento de clasificación, a cada individuo se le asigna un valor de adaptación. Luego, se utiliza un mecanismo de formación de nichos para evitar la convergencia prematura y mantener la diversidad.

Todos los elementos no-dominados de la población recibirán una clasificación de 1, mientras los dominados tendrán una clasificación > 1. En este esquema de clasificación, no todos los niveles están necesariamente representados en la población en una generación particular. Clasificados todos los elementos de la población, la asignación de *fitness* en MOGA se realiza de la siguiente manera:

1. Ordenar la población de acuerdo a la clasificación.

2. Asignar valor de adaptación a los individuos interpolando de la mejor clasificación a la peor de todas, de acuerdo a alguna función (usualmente lineal).

3. Promediar el valor de adaptación de los individuos con la misma clasificación, de tal forma que todos puedan ser muestreados a la misma tasa. Este procedimiento mantiene el *fitness* global de la población constante mientras que se mantiene una presión de selección apropiada.

Debido a errores aleatorios en el proceso de selección, cuando se tienen múltiples óptimos equivalentes, las poblaciones finitas convergen hacia una sola de ellas [FON93].

3.2.2. *Nondominated Sorting Genetic Algorithm*

El *Nondominated Sorting Genetic Algorithm* (NSGA) [DEB99], al igual que el MOGA, difiere del algoritmo genético simple sólo en la manera en que procede la selección, específicamente en la manera de asignar el valor de adaptabilidad. Los operadores de cruzamiento y mutación permanecen sin modificaciones. Así mismo, tanto el NSGA como el MOGA se basan en la utilización de un método de selección por clasificación para enfatizar los puntos actuales no-dominados y un método de *niching* [DEB99] para preservar la diversidad en la población. Sin embargo, el NSGA utiliza un procedimiento de clasificación que, a diferencia del utilizado en MOGA, no considera el número de elementos que dominan a un individuo para clasificarlo, sino su nivel de dominancia. Esto es, el NSGA utiliza en forma directa la idea de realizar un procedimiento de búsqueda de óptimos para optimización multiobjetivo basado en una clasificación según la no-dominancia, conforme fuera presentado en [DEB01].

3.2.3. *Niched Pareto Genetic Algorithm*

En [HOR93] se propone el *Niched Pareto Genetic Algorithm* (NPGA) el cual, como los anteriores, difiere del algoritmo genético simple únicamente en el procedimiento de selección. Sin embargo, este algoritmo no utiliza ningún procedimiento de clasificación por dominancia, sino que propone una variación del procedimiento de selección por torneo, llamado torneo por dominancia Pareto y la utilización de un procedimiento de *sharing* para romper empates y determinar un ganador.

En los procedimientos de selección por torneo un conjunto de elementos son seleccionados en forma aleatoria de la población genética actual y el mejor de este conjunto es seleccionado. Estos procedimientos, diseñados para EAs aplicados en problemas monobjetivo, asumen que se desea una única solución al problema. Para evitar que luego de cierto número de generaciones la población converja a una solución en un único punto, en [HOR93] se propone utilizar el operador de dominancia para la selección de individuos que compiten para ser seleccionados.

Si bien la relación de dominancia Pareto conduce de forma directa a un torneo binario, en el cual dos individuos seleccionados de manera aleatoria se comparan de acuerdo a la dominancia y aquel que domine al otro gana. Este tamaño de muestra es insuficiente para estimar que tan buena es una

solución con respecto al conjunto. Por ello en [HOR93] se propone un método distinto para la realización de la selección por torneo Pareto.

3.3. MOEAs de segunda generación

A pesar del éxito obtenido por los MOEAs de la primera generación basados en Pareto, una vez que estos encontraban una solución nodominada en una generación, éstas podían perderse cuando se aplicaban los operadores genéticos en las sucesivas generaciones. Para evitar la pérdida de buenas soluciones se aplica el concepto de elitismo.

Así, los MOEAs de segunda generación están caracterizados por la utilización de conceptos de Pareto conjuntamente con alguna forma de elitismo para la búsqueda de soluciones. Algunos algoritmos correspondientes a la segunda generación son: el *Strenght Pareto Evolutionary Algorithm* (SPEA) de Zitzler y Thiele [ZIT98], el NSGA2 de Deb, Agrawal, Pratab y Meyarivan [DEB00], el *Controlled Elitist NSGA-II* (CNSGA2) de Deb y Goel [DEB01], y el SPEA2 de Zitzler *et al* [ZIT02].

3.3.1. *Nondominated Sorting Genetic Algorithm II*

En [DEB00] se presentan los detalles del *Nondominated Sorting Genetic Algorithm II* (NSGA2), un nuevo algoritmo basado en clasificación por no-dominancia para asignar el valor de adaptabilidad a los elementos de la población genética. Son varias las características del NSGA2 que lo hacen diferente del NSGA original [SRI93, SRI94]. En primer lugar, el NSGA2 incorpora un mecanismo de preservación de elites que asegura el mantenimiento de las buenas soluciones encontradas con anterioridad. En segundo lugar, el NSGA2 utiliza un procedimiento rápido de clasificación por no-dominancia (*fastnondominated sorting procedure*) el cual incorpora un procedimiento especial de almacenamiento a fin de reducir la complejidad computacional del algoritmo presentado en [SRI93]. Por último, a diferencia de su antecesor, el NSGA2 no requiere de ningún parámetro ajustable [DEB00].

El procedimiento principal del NSGA2 se ilustra en la Figura 17.

Paso 1 Inicializar población P.
Paso 2 Ordenar P por no dominados.
Paso 3 Evaluar Individuos de P.
Paso 4 Aplicar operadores genéticos a P, para obtener P'
Paso 5 t=0 //t es cantidad de generaciones
Paso 6 R = P U P'.
Paso 7 Ordenar R considerando dominancia y obtener frentes F_i
Paso 8 I=1
Paso 9 Si no se cumple $|P_{i+1}| < N$ ir al Paso 14 // N es numero de individuos en P
Paso 10 Calcular la adaptabilidad de cada individuo en Fi
Paso 11 $P_{i+1} = P_{i+1} \cup F_i$
Paso 12 I = I +1
Paso 13 Volver al Paso 9
Paso 14 Elegir los primeros N elementos de P_{i+1}
Paso 15 Aplicar operadores genéticos a P_{i+1}, para tener P'_{i+1}
Paso 16 t=t+1
Paso 17 Mientras el criterio de parada no sea alcanzado volver al Paso 6.
Paso 18 Parar

Figura 17. Procedimiento principal del NSGA2

3.3.2. *Controlled Elitist Nondominated Sorting Genetic Algorithm II*

El algoritmo NSGA-II con elitismo controlado (*Controlled Elitist* NSGA-II – CNSGA2) [DEB01] difiere del NSGA2 únicamente en que el número de individuos pertenecientes al mejor frente actual de no-dominados es seleccionado de forma adaptativa. Durante el proceso evolutivo, en la población combinada del NSGA2, podrían existir demasiadas soluciones en el primer frente de no-dominados, las cuales pueden estar lejos del frente verdadero. Debido a la preponderancia de elementos en el primer frente de no-dominados, la información genética de los elementos pertenecientes a otros frentes puede perderse. En varios problemas, la pérdida de diversidad lateral puede producir una reducción en la velocidad de convergencia del EA, en especial en problemas multimodales donde las poblaciones pueden ser atraídas por frentes Pareto-óptimos locales [DEB01].

La Figura 18 detalla el procedimiento principal de este algoritmo.

Paso 1 Inicializar población P.
Paso 2 Ordenar P por no dominados.
Paso 3 Evaluar Individuos de P.
Paso 4 Aplicar operadores genéticos a P, para obtener P'
Paso 5 t=0 //t es cantidad de generaciones
Paso 6 $R_i = P_i \cup P'_i$.
Paso 7 F = ordenar por no dominados R_i
Paso 8 I=1
Paso 9 Si no se cumple $|P_{i+1}| < N$ ir al Paso 14 // N es numero de individuos en P
Paso 10 Asignar la distancia F_i
Paso 11 Ordenar F_i
Paso 12 $P_{i+1} = P_{i+1} \cup F_i [n_i:0]$ //se asignan los ni elementos de Fi
Paso 13 I = I +1
Paso 14 Volver al Paso 9
Paso 15 Aplicar operadores genéticos a P_{i+1}, para tener P'_{i+1}
Paso 16 t=t+1
Paso 17 Mientras el criterio de parada no sea alcanzado volver al Paso 6.
Paso 18 Parar

Figura 18. Procedimiento principal del cNSGA2

3.3.3. *Strength Pareto Evolutionary Algorithm*

En [ZIT98] se presenta un nuevo enfoque evolutivo para la optimización multiobjetivo, el *Strength Pareto Evolutionary Algorithm* (SPEA). Este algoritmo difiere de los anteriores en varios aspectos. Primeramente, utiliza dos poblaciones, incorporando el concepto de elitismo a través del almacenamiento de las soluciones no-dominadas en una población externa, la cual participa del procedimiento de selección. Además, el cálculo del valor de adaptación se realiza utilizando un procedimiento basado en la asignación de un valor de fuerza (*strength*) a todos los elementos de la población externa. Este procedimiento induce la formación de nichos a partir del concepto de dominancia Pareto, llamado *niching* por *strength* [ZIT98]. Puesto que el conjunto de soluciones en la población externa puede ser grande y ésta interviene en el proceso evolutivo, también se utiliza un procedimiento de *clustering* para reducir el número de soluciones en dicho conjunto.

En Figura 19 se observa la implementación del SPEA.

Paso 1 Generar una población inicial P y crear el conjunto no-dominado externo P' = 0.
Paso 2 Copiar los miembros no-dominados de P a P'.
Paso 3 Eliminar las soluciones en P' cubiertas por cualquier otro miembro de P'.
Paso 4 Si el número de soluciones en el almacenamiento externo excede un máximo N', reducir P' por medio de *clustering*.
Paso 5 Calcular el *fitness* de cada individuo en P, así como en P'.
Paso 6 Seleccionar individuos de P(t)+P'(t) (unión multiconjunto), hasta que el pool de apareamiento P(t +1) se llene.
Paso 7 Aplicar los operadores de mutación y cruzamiento específicos del problema como es usual.
Paso 8 t = t +1
Paso 9 Si se alcanza el máximo número de generaciones parar, sino ir al Paso 2.
Paso 10 Parar

Figura 19. Procedimiento principal del SPEA

3.3.4. *Strength Pareto Evolutionary Algorithm II*

El *Strength Pareto Evolutionary AlgorithmII* (SPEA2) se diferencia principalmente del SPEA, en que incorpora una estrategia fina de asignación del *fitness*, considera para cada individuo el número de los individuos que lo dominan y el número de los individuos por los cuales es dominado, y utilizando para la valoración de la densidad la técnica del "vecino más cercano" haciendo la búsqueda más eficiente [ZIT02].

En la Figura 20 observamos el pseudocódigo de este algoritmo.

Paso 1 Inicializar población P.
Paso 2 Evaluar Individuos de P.
Paso 3 Marcar soluciones no dominadas de P.
Paso 4 Actualizar el conjunto de soluciones no dominadas P_N
Paso 5 Calcular la adaptabilidad de los individuos de P y P_N
Paso 6 Seleccionar individuos del conjunto P_N
Paso 7 Aplicar los operadores de cruzamiento y mutación.
Paso 8 Mientras el criterio de parada no sea alcanzado volver al Paso 2.
Paso 9 Parar

Figura 20. Procedimiento principal del SPEA2

3.4. Algoritmos Evolutivos Paralelos Multiobjetivo

Una computadora paralela es un conjunto de procesadores capaces de trabajar cooperativamente para resolver un problema computacional. Esta definición es suficientemente amplia de forma a incluir supercomputadoras paralelas con cientos o miles de procesadores, conjuntos de estaciones de trabajo, estaciones de trabajo con varios procesadores, sistemas empotrados, etc. Las computadoras paralelas ofrecen la posibilidad de concentrar recursos computacionales en la solución de problemas computacionales importantes [FOS95].

Muchos algoritmos para la resolución de problemas son paralelizables por naturaleza. Aunque a veces es claro que la solución a muchos problemas puede obtenerse fácilmente ejecutando en forma simultánea algunas etapas del proceso de resolución, una conversión sencilla de un algoritmo secuencial a un algoritmo paralelo puede no proveer el máximo paralelismo obtenible para un problema dado. El hecho es que convertir el mejor algoritmo secuencial en el mejor algoritmo paralelo no es trivial, esto hace necesario el estudio y desarrollo de técnicas de ingeniería para la construcción y el diseño de algoritmos paralelos [FOS95].

La integración de técnicas de computación paralela con otras tecnologías permite el desarrollo de nuevas aplicaciones comerciales y científicas. En particular, la integración de la computación paralela y la computación evolutiva ha dado nacimiento a los algoritmos evolutivos paralelos multiobjetivo (*parallel Multiobjective Evolutionary Algorithms* - pMOEAs).

Los algoritmos evolutivos multiobjetivo han sido utilizados para la resolución de un gran número de MOPs como la optimización de potencia reactiva [BAR01], diseño de redes de computadoras [DUA01], entre otros. Con el creciente interés en la utilización de MOEAs en problemas de optimización del mundo real, se hace necesario mejorar el desempeño de estos algoritmos tanto en la calidad de las aproximaciones obtenidas como en la velocidad del proceso de búsqueda. La utilización de conceptos de paralelismo en MOEA es una alternativa para conseguir estas mejoras, puesto que la utilización de un número mayor de procesadores (y memoria) trabajando en el problema, en principio, permite la exploración de un espacio de soluciones mayor en un periodo dado de tiempo [PVM94].

La utilización de pMOEAs, para la resolución de problemas de optimización multiobjetivo, posee varias ventajas con respecto a otros métodos, uniendo las características propias de los MOEAs

con las ventajas del cómputo paralelo. Básicamente, los pMOEAs buscan encontrar soluciones tan buenas o mejores que sus contrapartes secuenciales en menos tiempo y/o explorar un espacio mayor de posibles soluciones [FLY95]. Es decir, siendo fácilmente paralelizables, los pMOEAs ofrecen al menos una de las siguientes ventajas adicionales, con relación a otras técnicas de resolución de MOPs [FLY95]:

- capacidad de explorar espacios de búsqueda complejos y de alta dimensionalidad;
- capacidad de explorar espacios de búsqueda mayores que sus contrapartes secuenciales;
- posible reducción del tiempo total de ejecución;
- posible mejora en la calidad de las soluciones obtenidas.

En la actualidad existen varios modelos propuestos para el desarrollo de EAs paralelos. Los tres modelos más importantes son:

- el modelo maestro-esclavo (*master-slave model*),
- el modelo de difusión (*diffusion model*)
- el modelo de islas (*island model*).

3.4.1. Modelo maestro-esclavo

Para la mayor parte de los problemas multiobjetivo, el principal costo computacional viene dado por el cálculo de funciones objetivos complejas, las cuales pueden incluir, por ejemplo, la realización de un proceso de simulación como en [DUA01]. Entonces, paralelizar el cómputo de esta función es una alternativa para obtener una mejora en el desempeño del algoritmo. Los esclavos realizan el cálculo de la función objetivo y el maestro se encarga de distribuir el cómputo entre los esclavos así como de la ejecución de los operadores evolutivos.

Al acelerar el cómputo de la función objetivo, el modelo maestro-esclavo posibilita la realización de más generaciones que un algoritmo secuencial en el mismo tiempo. Así, el modelo maestro esclavo permite la exploración de un espacio de búsqueda mayor en un tiempo dado, por lo que es de esperar que se obtengan mejores aproximaciones que utilizando un algoritmo secuencial. Sin embargo, aunque con este modelo es posible obtener una mejora en la velocidad de ejecución,

el comportamiento global del algoritmo evolutivo multiobjetivo paralelo maestro-esclavo es el mismo que el de los algoritmos secuenciales.

La distribución de la evaluación de la función objetivo sobre un conjunto de procesadores esclavos generalmente se implementa de tres formas diferentes [VEL03]:

1. Distribuyendo los miembros de la población entre los esclavos, donde cada esclavo realiza la evaluación de las k funciones objetivo.
2. Distribuyendo miembros de la población entre un conjunto de k esclavos, donde cada uno realiza la evaluación de uno de los objetivos.
3. Distribuyendo el cálculo de la función objetivo en sí misma para toda la población entre varios procesadores.

En el primer método, cada procesador esclavo calcula todos valores de los objetivos de los individuos que se le asignan. Cuando este método se implementa sobre un sistema homogéneo, puesto que todos los esclavos computan funciones objetivo idénticas, para un número idéntico de soluciones, los esclavos usualmente terminan su ejecución en un tiempo similar [VEL03].

En el segundo método, las k funciones objetivos se mapean cada una a un esclavo. Puesto que la complejidad de las funciones puede variar, este método requiere una adecuada distribución de las k funciones a los distintos procesadores. El balance de carga puede ser de utilidad para distribuir adecuadamente el costo computacional entre los distintos procesadores utilizados, sin embargo, el esfuerzo para esto podría exceder cualquier ganancia final [VEL03].

El último método distribuye el cálculo de la función objetivo en si misma, entre múltiples procesadores.

Cada una de las funciones objetivo es particionada para su resolución entre múltiples procesadores. Este método es adecuado para problemas realmente grandes.

3.4.2. Modelo de Difusión

En busca de una mejora en el desempeño, otra alternativa para el desarrollo de algoritmos evolutivos paralelos es la paralelización de los distintos operadores evolutivos. Puesto que estos operadores son relativamente sencillos, en general, la relación entre el costo de la comunicación y

el costo computacional hace que la aceleración que se pueda obtener, cuando existe, sea escasa. Sin embargo, el modelo de difusión propone un esquema de selección distribuida de elementos pertenecientes a distintas subpoblaciones.

Como en el modelo de maestro-esclavo, en el modelo de difusión se considera conceptualmente una única población donde en cada procesador se tienen sólo unos pocos individuos. Este modelo supone que el cruzamiento entre ciertas subpoblaciones llevará a buenas soluciones, las cuales se encuentran en diferentes áreas, distribuidas convenientemente cuando se considera toda la población.

Por tanto, este modelo precisa la imposición de alguna estructura que restrinja la selección y recombinación entre elementos que se encuentran en ciertas subpoblaciones. Debido a que la selección de individuos se realiza entre distintas subpoblaciones, el costo de la comunicación puede ser alto en este modelo.

3.4.3. Modelo de Islas

Cuando se utilizan sistemas de cómputo fuertemente acoplados, puesto que la relación del tiempo de cómputo respecto al tiempo de comunicación es alta, la paralelización del cálculo de la función objetivo o del procedimiento de selección son alternativas válidas para conseguir la aceleración de la ejecución. Sin embargo, en países como el Paraguay, este tipo de máquinas paralelas representa un costo prohibitivo para la mayor parte de las empresas e instituciones de enseñanza e investigación. Por lo tanto, la utilización de un conjunto de estaciones de trabajo como una máquina paralela (COW), ya sea conectadas a una red de área local (*Local Area Network* - LAN) o conectadas a través de Internet, se vuelve la única alternativa posible.

En estos entornos el costo de la comunicación es elevado, siendo deseable una granularidad mayor en el paralelismo. Además, es necesario considerar otras cuestiones, como la posible pérdida de una conexión, heterogeneidad de procesadores, etc. Una posibilidad de paralelización de fácil adaptación al entorno computacional señalado es la ejecución simultánea de varios MOEAs en procesadores diferentes y que sus resultados se comparen, contrasten o combinen. A este esquema de paralelización se le denomina modelo de islas.

El modelo de islas está basado en el fenómeno natural de que las poblaciones se encuentran, usualmente, relativamente aisladas unas de otras. Los pMOEAs basados en este modelo se denominan MOEAs de multipoblación o distribuidos. La característica principal de los pMOEAs basados en el modelo de islas es que los individuos de una población particular pueden migrar de forma ocasional a alguna otra.

Conceptualmente, en el modelo de islas, una población se divide en un número de poblaciones separadas e independientes. Los diferentes operadores evolutivos trabajan en cada isla, lo que implica que las distintas poblaciones se encuentran buscando en regiones posiblemente diferentes del espacio de búsqueda.

Cada isla también podría tener distintos parámetros así como diferente estructura de MOEA. Además, individuos de una isla podrían migrar a otra isla de acuerdo a algún criterio.

Los distintos procesos que intervienen en la búsqueda de soluciones pueden comunicarse entre sí utilizando distintas topologías de interconexión, tanto lógicas como físicas.

El modelo de islas requiere la selección de políticas de migración que señalen entre otras:

- la manera en que los individuos migrarán,
- el número de migrantes,
- la frecuencia de migración,
- de donde se seleccionarán los elementos a migrar y como se realizará el reemplazo de los elementos
- en una población por los migrantes provenientes de otras poblaciones,
- los distintos parámetros y algoritmos a utilizar en cada una de las diferentes islas.

Definiendo de forma conveniente la política de migración, este modelo puede aplicarse a varias arquitecturas paralelas, especialmente en sistemas paralelos de memoria distribuida.

Existen básicamente cuatro variantes de pMOEAs basados en el modelo de islas [VEL03]:

1. homogéneos: en todas las islas se ejecutan el mismo MOEA con parámetros idénticos;
2. heterogéneos: en las distintas islas se ejecutan distintos MOEAs o el mismo MOEA con distintos parámetros;
3. cada isla evalúa un conjunto de funciones distinto;
4. en cada isla se representa una región diferente del dominio fenotípico o genotípico.

Capítulo 4

Algoritmos en Equipo

Cuando no se dispone de un método único de resolución, se estudian diversos métodos de optimización, cada uno con sus propias características, que lo hacen apropiado para un subconjunto de problemas. En algunos casos, un método tiene un buen desempeño bajo ciertas características operativas, pero falla para otros escenarios del problema para los cuales existen alternativas diferentes de solución, que a su vez sólo son aplicables a una determinada clase de problemas. Surge así la necesidad de utilizar un método de optimización adecuado para cada clase de problema. Ocasionalmente, puede suceder que ninguno de los métodos disponibles consigue resolver adecuadamente el problema en cuestión, cuando son utilizados aisladamente [BAR96, BAR98].

Ante esta dificultad, surge la posibilidad de descomponer un problema complejo en subproblemas menores, de forma a utilizar métodos diferentes en cada uno de estos subproblemas, según sean sus características específicas. Esta combinación de algoritmos diferentes interactuando en la solución de un mismo problema global, se conoce como *Team Algorithm* [BAR96, BAR98].

La solución de problemas utilizando *Team Algorithm* puede ser naturalmente implementada en paralelo, asignando los diferentes subproblemas a los diversos procesadores de un sistema distribuido asíncrono, como una red de computadoras, en cuyo caso se lo conoce como *A-Team* (*Asynchronous Team*) [BAR95a, BAR96, BAR98].

4.1. Historia de los *Team Algorithms*

Pensar en la posibilidad de combinar diferentes métodos numéricos con el fin de obtener un algoritmo que sea capaz de optimizar grandes problemas superando las desventajas que presentan al aplicarlos por separado resalta el trabajo de Dusonchet *et al*. [DUS71], quienes en 1971 desarrollaron

uno de los primeros trabajos en el que se combinan diferentes algoritmos en la resolución de problemas de ingeniería, tales como el problema del flujo de potencia eléctrica, combinando métodos numéricos conocidos, tales como el método de Jacobi y el método de Newton, presentando resultados positivos. En este trabajo ya se presentaba la idea de dividir un sistema algebraico de ecuaciones grande, constituido por un gran número de variables, en varios sistemas de ecuaciones más pequeños, de modo que sobre cada uno de ellos se aplique un método numérico adecuado a las características propias del subconjunto de ecuaciones a resolver. De este modo, con el trabajo de Dusonchet *et al*. comienza la motivación para el desarrollo de los *Team Algorithms*, especialmente en lo que se refiere a dividir un problema en subproblemas menores y aplicar a cada subproblema un método numérico adecuado para resolverlo. A partir del trabajo de Marschak y Radner [MAR72] se comienza a mencionar la idea de combinar diferentes algoritmos para la resolución de un mismo problema global. Nace así la posibilidad de usar estos algoritmos en sistemas distribuidos, que son sistemas computacionales formados por un conjunto de procesadores con capacidad de intercambiar mensajes o datos. Por esta razón, a partir del mencionado trabajo, y durante toda la década de los 80 [BAR93], comienzan los *Team Algorithm* a ser estructurados como una combinación de diferentes algoritmos (o métodos) aplicados a cada procesador de un sistema distribuido; como por ejemplo, el trabajo desarrollado en 1982 por Talukdar, Pyo y Giras [TAL82] y su extensión en Talukdar, Pyo y Mehrotra en 1983 [TAL83]. En dichos trabajos, se desarrolló un *Team Algorithm* (ilustrado en la Figura 21) formado por una combinación de diferentes métodos, contando además con un *Coordinador*, que consiste en un proceso que controla y supervisa la comunicación entre los diversos métodos que están siendo ejecutados en los diferentes procesadores de un sistema distribuido. El algoritmo anterior fue aplicado a la resolución de sistemas de ecuaciones algebraicas, formados por una gran cantidad de variables, demostrándose así que dicho algoritmo es capaz de resolver problemas que ninguno de los métodos combinados consigue resolver cuando son aplicados en forma independiente.

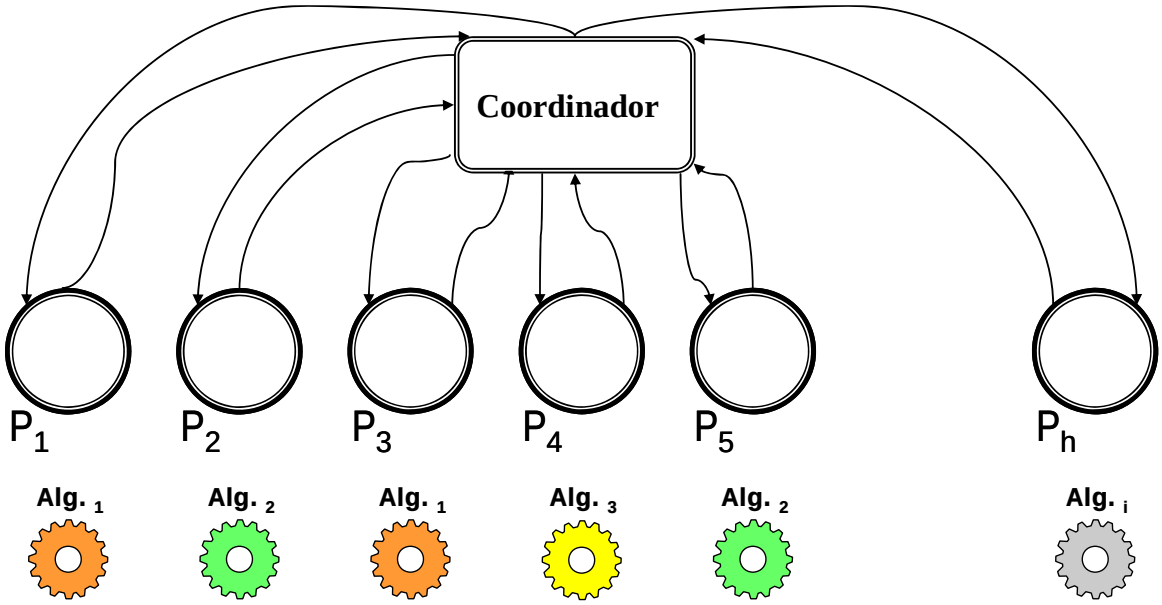

Figura 21. *Team Algorithm*

En el trabajo realizado por Talukdar et al. [SOU91] en 1991, se desarrolló un *A-Team* en el que se combinó el Algoritmo Genético, que es una técnica de Inteligencia Artificial, con un método numérico conocido, tal como el método de Newton, aplicado a la resolución de sistemas de ecuaciones algebraicas no lineales. Los resultados obtenidos en este trabajo fueron muy promisorios confirmando las ventajas de utilizar los *ATeams* en complejos sistemas de ecuaciones matemáticas.

A su vez, en el trabajo desarrollado por Talukdar, Ramesh y Nixon [TAL91] se estableció una analogía entre los *A-Teams* y una sociedad de insectos, basado en el trabajo de Fox [FOX81], que postula la posibilidad de obtener un efecto sinergético, es decir, mostrar que la combinación de algoritmos es más que una simple suma de propiedades de cada uno de los algoritmos combinados. En estos trabajos también se describe la forma de aplicar los *A-Teams* en sistemas de administración de energía (*Energy Managements Systems*), control de redes eléctricas y otras aplicaciones relacionadas con sistemas de potencia.

Una discusión más completa puede ser encontrada en Ramesh *et al.* (1991) [RAM91], en la cual los *A-Teams* son utilizados en aplicaciones tan diversas como: soluciones de sistemas de ecuaciones algebraicas no lineales, combinación de Algoritmos Genéticos (GA) con métodos de cálculo intensivo, proyecto de edificios, control de redes eléctricas y otras aplicaciones de sistemas de potencia. Más tarde, Baran [BAR93] estudió formalmente la convergencia de los *A-Teams*.

En la actualidad, los *A-Teams* han sido aplicados en numerosos problemas difíciles e importantes [BAR98], tales como: diseño de rascacielos, diagnósticos de fallas en redes eléctricas, control de redes eléctricas, problema de transporte de aceite a través de tuberías, aplicación de *A-Teams* a la industria del acero, generación de energía, entre otros.

4.2. Concepto de los *Asynchronous Teams*

La idea para aprovechar eficientemente un sistema distribuido consiste en la descomposición de un problema complejo en subproblemas menores, asignando luego cada uno de ellos a un número de procesadores disponibles de un sistema distribuido de forma tal que cada procesador resuelva uno o más subproblemas cuidando que el conjunto resuelva el problema global. Para la resolución de cada subproblema se puede elegir el método más adecuado, lo que lleva a tener una combinación de métodos variados siendo ejecutados en unidades de procesamiento diferentes y comunicando los resultados parciales de tal forma que todos colaboran para la resolución global del problema (*Team Algorithm*) [BAR96].

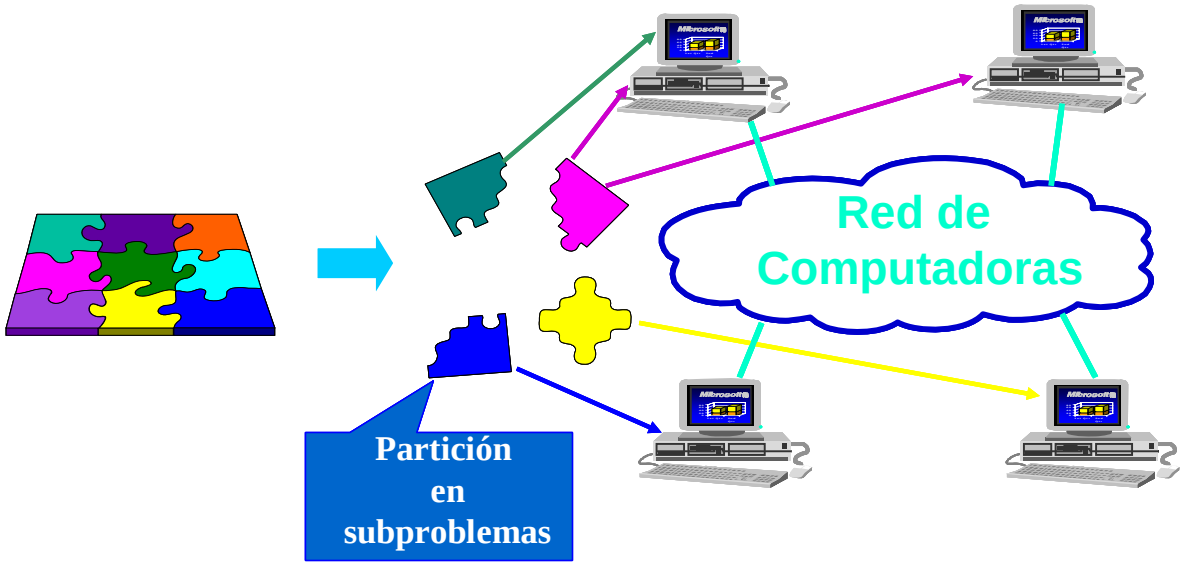

Figura 22. Implementación del *A-Team*

Los algoritmos adecuados a cada subproblema a resolver, y que son asignados a los diferentes procesadores disponibles del sistema distribuido, pueden ser diseñados de tal forma que la

comunicación entre dichos procesadores no esté sincronizada, permitiendo de esta forma aprovechar al máximo los recursos computacionales en redes de computadoras posiblemente heterogéneas (diferentes desempeños, recursos y velocidades de procesamiento) sin necesidad de invertir en recursos computacionales costosos para la sincronización de tareas. Este tipo de implementación paralela se conoce en la literatura como *A-Teams* (*Asynchronous Teams*) [BAR96] y la podemos observar en la Figura 22.

El número de procesadores disponibles en la red, al igual que el número de algoritmos utilizados puede ser arbitrariamente grande. Además, dichos algoritmos pueden ser distribuidos sobre un área de comunicación arbitrariamente grande. Cada algoritmo puede ser completamente autónomo (cada algoritmo decide con qué resultados trabajará y cuando).

De esta forma, los *A-Teams* han sido considerados como extremadamente eficientes para la resolución de problemas muy difíciles para los cuales muchos métodos tradicionales utilizados en forma independiente no tienen un desempeño satisfactorio. Algunos métodos pueden ser muy precisos en sus resultados pero a la vez son muy lentos, otros son rápidos en el procesamiento pero muy poco precisos en la solución obtenida. Ocasionalmente, puede suceder que ninguno de los métodos disponibles consigue resolver el problema en cuestión, cuando son utilizados en forma independiente [BAR96].

Así, un *A-Team* combina estos algoritmos de forma a obtener un mejor desempeño en la resolución de problemas complejos, mejorando la calidad de los resultados obtenidos en comparación a los resultados hallados por los métodos tradicionales, aumentando además la velocidad de procesamiento.

4.3. TA-MOEA. Modelo Utilizado

El desempeño de un algoritmo se mide generalmente por la calidad de los resultados obtenidos y por la rapidez en la respuesta. En este aspecto, los *Team Algorithms* han demostrado ser muy eficientes superando muchas veces en desempeño a los métodos tradicionales [BAR95a, BAR95b, BAR96, BAR98]. Debido a estas ventajas en combinar diferentes métodos, se realizó en el presente trabajo la combinación de los siete Algoritmos Evolutivos Multiobjetivos presentados en el capítulo 3

implementados en forma paralela y fueron aplicados a los seis problemas ZDT propuestos en el capítulo 2.

Como ya se ha visto, una alternativa para disminuir el tiempo de computación consiste en paralelizar el problema, descomponiéndolo en numerosos subproblemas. Al respecto, los *Team Algorithm* han surgido como una novedosa herramienta que no solo mejora los tiempos de respuesta del algoritmo sino que se ha comprobado que tiene un efecto sinergético [BAR95a, BAR95b, BAR96, BAR98], esto es, el algoritmo no solo mejora la velocidad de procesamiento sino también la calidad de los resultados, razón por la cual en el presente capítulo se describe el modelo de un *Team Algorithm* formado por un proceso Coordinador y siete procesos paralelos esclavos en los cuales se pueden ejecutar uno de los siete MOEAs disponibles. El modelo utilizado se denomina entonces *Team Algorithm* of pMOEAs y se ilustra en la Figura 23.

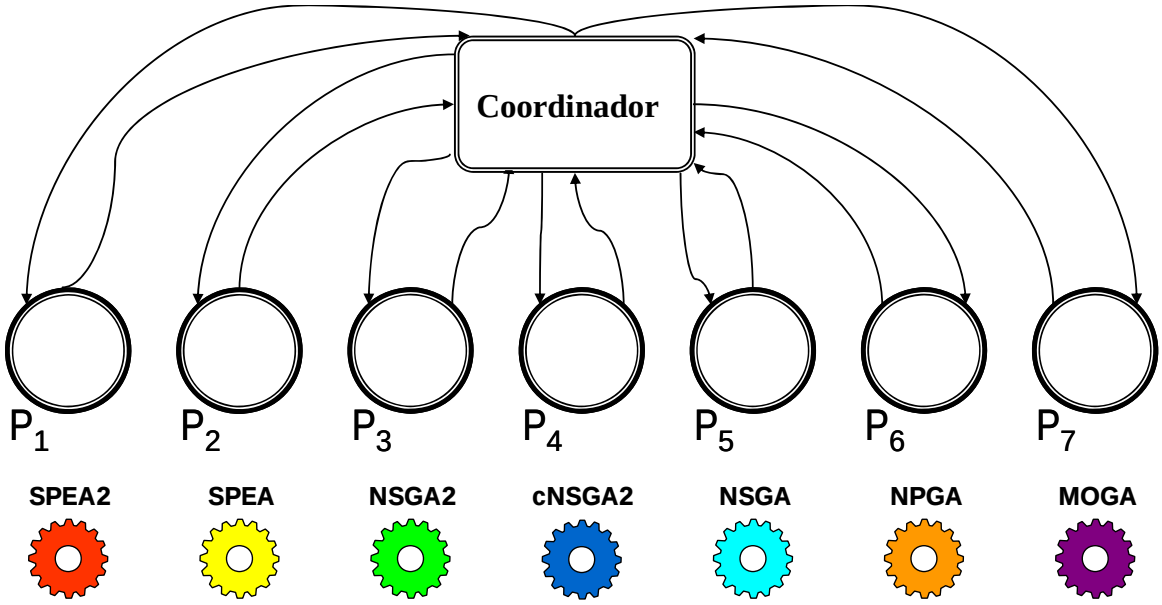

Figura 23. Modelo del *Team Algorithm* of pMOEAs

El modelo propuesto se basa en la utilización de dos tipos distintos de procesos, formando un equipo de trabajo:

- un proceso Coordinador;
- siete procesos pMOEAs.

El proceso diagrama de flujo del proceso Coordinador se observa en la Figura 24. El Coordinador se encarga de seleccionar, iniciar y controlar las distintas instancias de los MOEAs utilizados y de almacenar soluciones obtenidas por éstos.

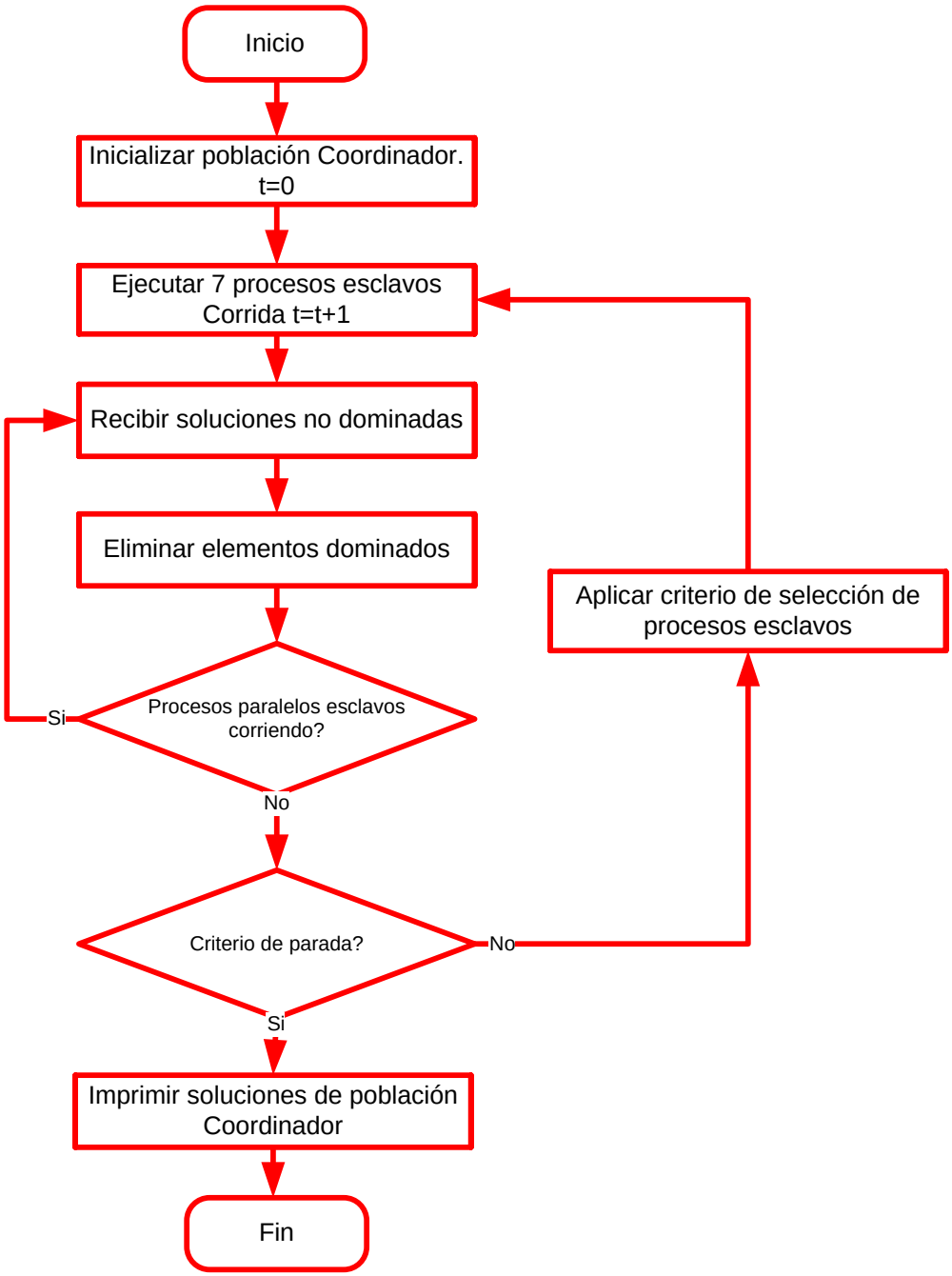

Figura 24. Diagrama de Flujo del proceso coordinador

A los distintos MOEAs que realizan la búsqueda de soluciones se les ha agregado un procedimiento aleatorio de migración y uno de recepción asíncrona de soluciones.

El proceso Coordinador primeramente lee el número de procesos MOEA que trabajaran en la búsqueda de soluciones. Puesto que cada uno de éstos puede corresponder a un tipo de MOEA distinto o bien al mismo tipo de algoritmo evolutivo pero con diferentes parámetros, el Coordinador también precisa información sobre el tipo de MOEA que se utilizará así como los parámetros que le corresponden, para inicializarlos posteriormente en forma conveniente. Así mismo, recibe la especificación del número de soluciones deseadas. Seguidamente crea las estructuras para almacenar los resultados provenientes de los distintos procesos MOEAs en la población Coordinador. Luego, el Coordinador se agrega a un grupo de trabajo e inicia cada uno de los procesos MOEAs con sus parámetros específicos. Esto puede ser implementado utilizando primitivas de comunicación de grupo proveídas por librerías de paso de mensajes [PVM94], lo que facilita la comunicación entre los distintos procesos utilizados.

Como se explica más adelante, durante la evolución, cada uno de los MOEAs que interviene en la búsqueda envía a todos los elementos que componen el equipo de trabajo un porcentaje de las mejores soluciones obtenidas. Cuando el Coordinador recibe estas soluciones las almacena en la población Coordinador. A fin de mantener sólo las mejores soluciones, se eliminan las soluciones cubiertas o dominadas.

Si los resultados recibidos corresponden a los de la última generación de un proceso MOEA, se reduce la cuenta de MOEAs en ejecución. Cuando ésta cuenta es igual a cero, se procede a eliminar las soluciones cubiertas existentes en la población Coordinador y a la escritura final de los resultados obtenidos.

El Algoritmo representado en la Figura 25 presenta el marco general propuesto para la implementación de los distintos pMOEAs. Al comienzo, se realizan algunas tareas propias de inicialización. Primeramente, se reciben desde el Coordinador los distintos parámetros del MOEA utilizado. Además de los parámetros usuales, probabilidad de cruzamiento, selección, mutación, radio de nicho, etc., también se reciben la probabilidad de migración (p_{mig}) así como el número máximo de elementos nodominados a migrar (n_{mig}). Tras la fase de recepción de los parámetros del algoritmo, cada pMOEA se agrega a su equipo de trabajo.

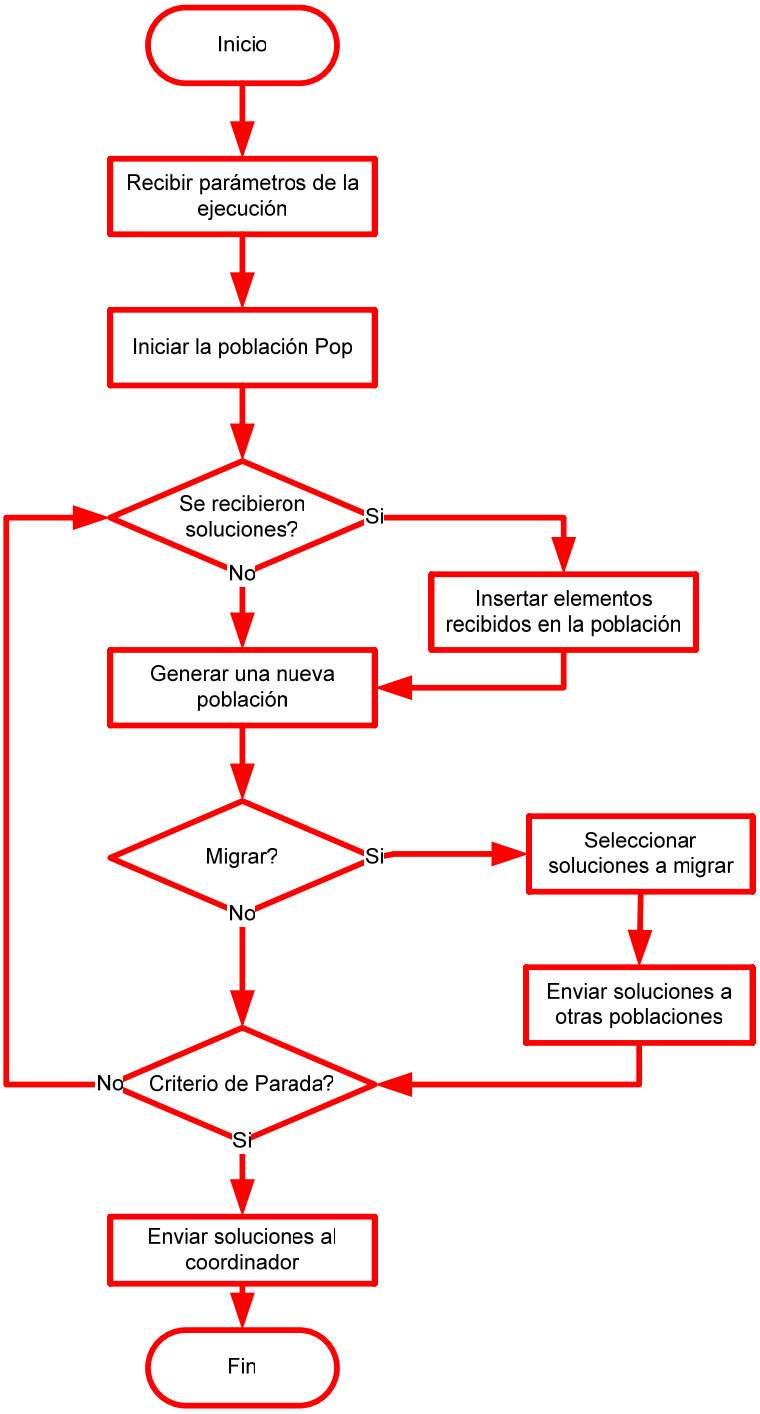

Figura 25. Diagrama del pMOEA

Luego de la inicialización, los pMOEAs realizan el proceso evolutivo. En cada generación, se determina si se han recibido elementos provenientes de otras subpoblaciones. Si este es el caso, se

realiza la recepción de los mismos. Luego de la recepción de los elementos se procede al reemplazo de los elementos dominados por los recibidos. En el peor caso, ninguna solución en el Coordinador es dominada por alguna solución recibida. Cuando esto ocurre, el procedimiento de reemplazo no tiene ningún efecto sobre la estructura de la población genética, aunque si un efecto negativo sobre la velocidad del algoritmo. Posteriormente, se realizan los procedimientos usuales de evolución para generar una nueva población. Luego, se determina si corresponde la migración de elementos en esa generación *t*. Si corresponde la migración, se obtienen los elementos no dominados existentes en $P(t)$ para el caso de los algoritmos sin población externa y luego se selecciona de entre ellos, aleatoriamente, como máximo n_{mig} individuos, los cuales son enviados a todos los procesos que componen el equipo de trabajo. Transcurrido un cierto número de generaciones o alguna otra condición *a priori*, se envían todos los elementos de $\boldsymbol{Y}_{known}(t)$ al proceso Coordinador. Al alcanzar un número máximo de generaciones, el pMOEA envía todas sus soluciones al Coordinador y finaliza.

El modelo propuesto garantiza que recibida una solución, en una de las islas, ésta será aceptada para formar parte de la población sólo si domina a alguna solución en el conjunto. Note que no se utiliza ningún procedimiento de almacenamiento o control sobre el conjunto de soluciones que ya han sido enviadas. Si bien tal procedimiento podría ser beneficioso para evitar el envío repetitivo de soluciones ya encontradas por un procesador, esto adhiere complejidad adicional al pMOEA en la etapa de envío, lo que podría no ser correspondido por una mejora en la eficiencia del algoritmo. Además, puesto que es de esperar que los envíos se produzcan luego de cierto número de generaciones y que el conjunto de soluciones no dominadas cambie y por tanto se modifique el conjunto de soluciones a enviar, al menos parcialmente. Por otro lado, las soluciones a migrar son seleccionadas de forma aleatoria entre el conjunto de soluciones no dominadas y enviadas de forma asíncrona, por lo que aún cuando por azar se repita el envío de soluciones, la recepción de ellas no está garantizada. Es más, aunque se garantice la recepción, no se garantiza que el receptor acepte la solución pues podría tratarse de una solución dominada con respecto a la población del receptor.

4.4. Implementación Estática

En las implementaciones estáticas, el Coordinador tiene fijos los siete procesos pMOEA y los mantiene hasta que el criterio de parada sea alcanzado.

Después de experimentar un cierto tiempo con la combinación de algoritmos a ser utilizada, se seleccionó la configuración detallada a continuación.

4.4.1. TA-MOEA S

El TA-MOEA S es el *Team Algoritm of MOEAs Static*, es decir, un Team Algorithm con algoritmos seleccionados de antemano y que se mantienen hasta el final.

Como se observa en la Figura 26, los siete procesos de algoritmos que utiliza el TAMOEA-S son: dos procesos SPEA2, dos procesos SPEA, dos procesos NSGA2 y un proceso cNSGA2.

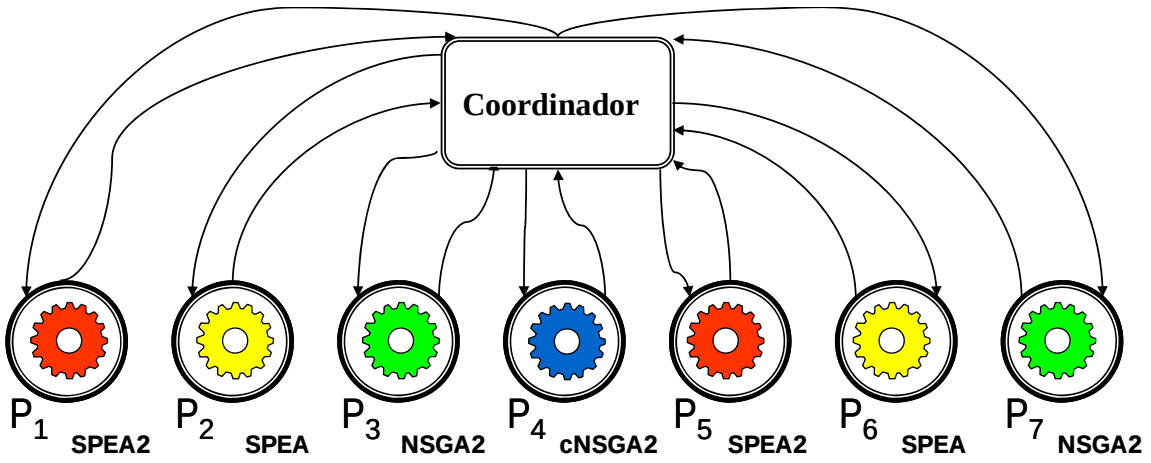

Figura 26. TA-MOEA S

Los procesos Coordinador y pMOEAs se implementan siguiendo la Figura 24 y Figura 25 respectivamente. El proceso Coordinador es muy simple y solo se encarga de administrar a los procesos paralelos esclavos.

4.5. Implementaciones Dinámicas

En las implementaciones estáticas, el Coordinador va seleccionando los siete procesos pMOEA y los va cambiando siguiendo una política hasta que el criterio de parada sea alcanzado.

En el proceso Coordinador resulta crítica la forma de determinar que algoritmo está produciendo mejores resultados que otro. Para poder establecer esto se adoptó en este trabajo el conteo de las soluciones no dominadas que introducen los algoritmos en la población del Coordinador, esto es, el algoritmo que introduce mayor cantidad de soluciones en la población Coordinador será considerado el mejor para ese problema. De este modo se puede establecer un *ranking* de los algoritmos que corren en los procesos paralelos esclavos.

Se proponen dos TA-MOEAs dinámicos, el primero basado en probabilidades y el segundo basado en el concepto de elitismo.

4.5.1. TA-MOEA P

El TA-MOEA P es el *Team Algoritm of MOEAs Probabilistic*, es decir, un *Team Algorithm* con algoritmos seleccionados de forma probabilística.

Los procesos Coordinador y pMOEAs se implementan siguiendo los delineamientos de la Figura 27 y la Figura 25 respectivamente. El proceso Coordinador Probabilístico se encarga de elegir y administrar a los procesos paralelos esclavos.

La probabilidad de elección del algoritmo Alg_i se denomina p_i y para nuestro modelo en particular se determina por:

$$p_i = \frac{PC_i + 1}{PC_T + 7} \qquad \textbf{(4.1)}$$

siendo PC_i el número de soluciones no dominadas aportadas por el Alg_i a la población del Coordinador y PC_T el número total de soluciones que posee la población del Coordinador.

Con esta propuesta, por más que el algoritmo no aporte ninguna solución a la población del Coordinador, igual mantiene una pequeña probabilidad de ser elegido.

Evidentemente mientras más soluciones haya aportado el algoritmo Alg_i a la población del Coordinador, sus probabilidades aumentan al incrementarse su porcentaje de selección y recíprocamente mientras menos soluciones haya contribuido, sus posibilidades de elección disminuyen.

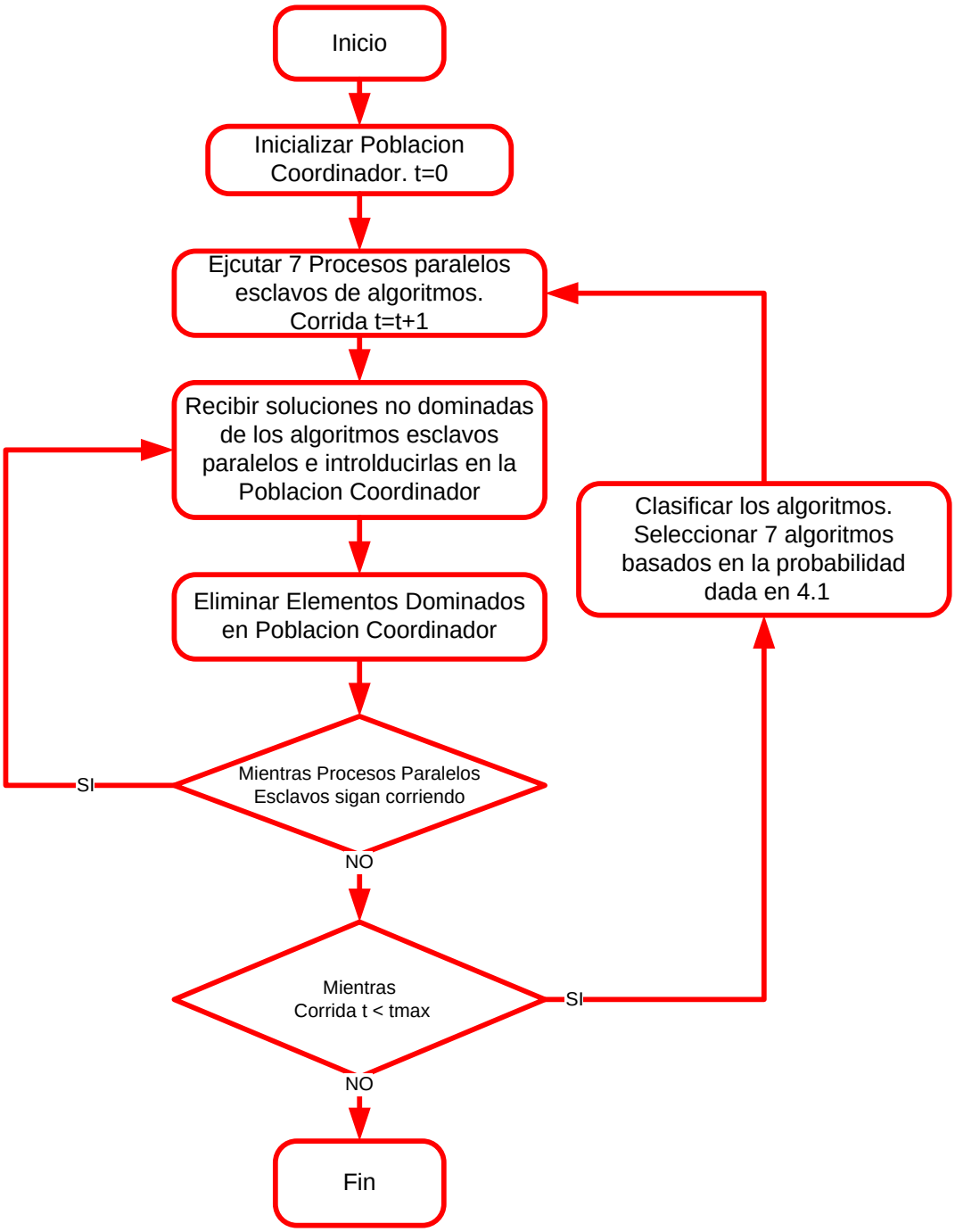

Figura 27. Diagrama de Flujo del Coordinador Probabilístico

4.5.2. TA-MOEA E

El TA-MOEA E es el *Team Algoritm of MOEAs Elitist*, es decir, un *Team Algorithm* con algoritmos seleccionados de forma elitista.

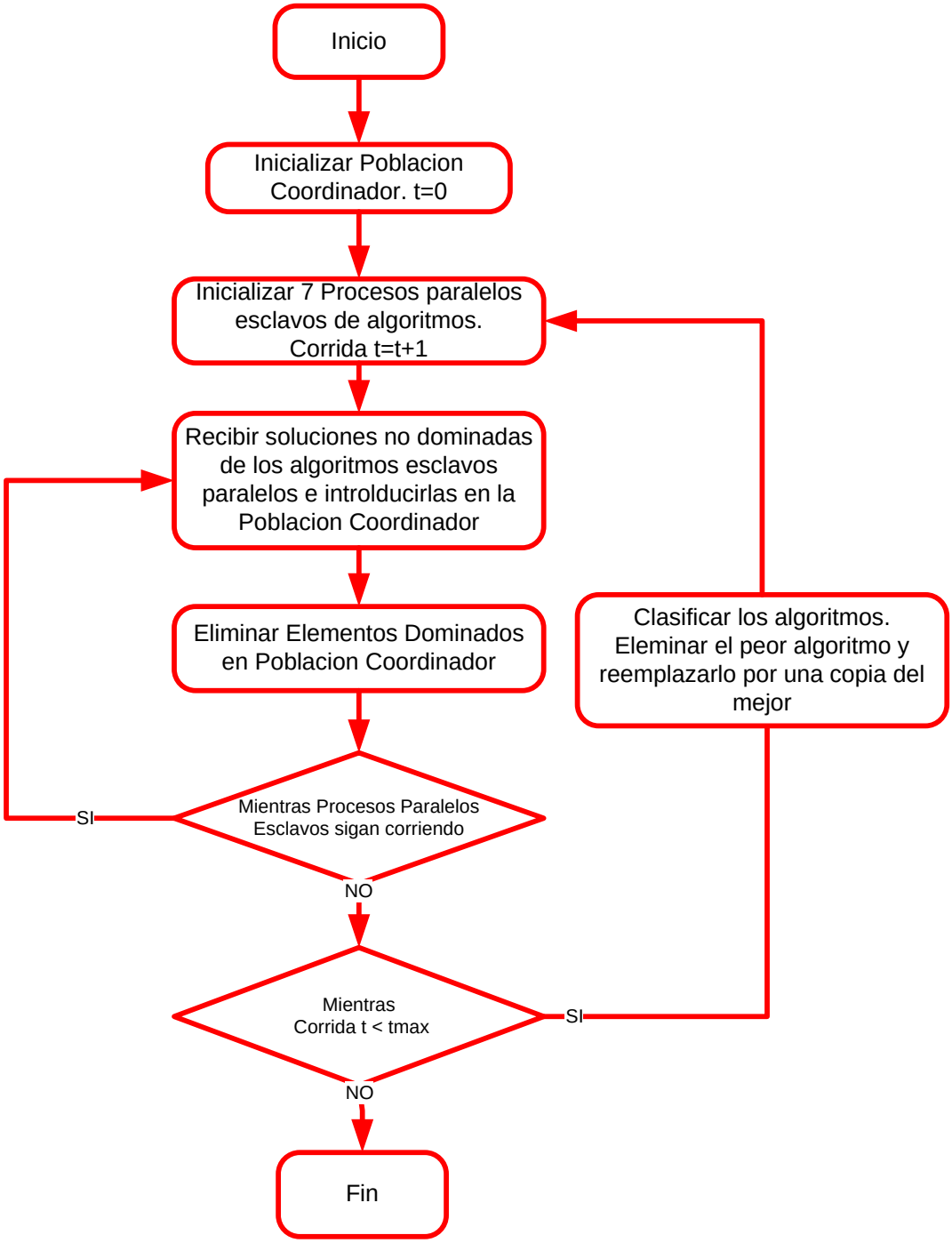

Figura 28. Diagrama de Flujo del Coordinador Elitista

Los procesos Coordinador y pMOEAs se implementan siguiendo los delineamentos de la Figura 28 y la Figura 25 respectivamente. El proceso Coordinador Elitista se encarga de elegir y administrar a los procesos paralelos esclavos.

El proceso de elección establece un ranking de los algoritmos basado en el número de soluciones no dominadas aportadas a la población Coordinador. El peor algoritmo es descartado y en su lugar se ubica una copia del mejor. Para ejemplificar esto, se puede suponer que en un momento dado los siete procesos paralelos esclavos y su clasificación sean los siguientes:

1. NSGA2
2. SPEA2
3. SPEA2
4. NSGA2
5. cNSGA2
6. SPEA
7. NPGA

Entonces descartando el peor (el NPGA) y duplicando el mejor (NSGA2) para la siguiente iteración tendremos tres procesos NSGA2, dos SPEA2, un cNSGA2 y un SPEA.

Esto nos da la pauta que ha medida que transcurran las iteraciones el conjunto de procesos tiende a ejecutar el mismo algoritmo, y dicho algoritmo será el mejor del conjunto aplicado al problema específico que se está resolviendo. Por lo tanto, un aporte adicional que surge del TA-MOEA E es la capacidad de conocer razonablemente cual es el mejor MOEA del conjunto disponible en una biblioteca para un determinado problema.

Capítulo 5

Resultados Experimentales

En el presente Capítulo se presentan los experimentos realizadas con el objeto de evaluar los TA-MOEAs propuestos.

Primero se presentan las distintas métricas utilizadas en la comparación de los seis problemas de prueba seleccionados. Así mismo, se presenta el análisis de los resultados obtenidos por los distintos conjuntos de algoritmos seleccionados a la luz de las distintas métricas y sobre los problemas considerados.

5.1. Métricas Utilizadas

La elección de las métricas que sean capaces de dar valores adecuados para el desempeño de los *MOEA*s en diferentes aspectos, y por ende permitir comparaciones significativas entre diferentes algoritmos e implementaciones es todavía un ámbito de discusión e investigación. En tal sentido gran parte de las métricas propuestas se han aplicado solamente a problemas que no provienen del mundo real, sino que se han diseñado para propósitos teóricos y de pruebas de laboratorio. Queda aún abierto el tema de escoger cuáles de estas métricas se pueden aplicar exitosamente a problemas reales y qué tan significativos pueden ser los resultados obtenidos con ellas.

La selección de las métricas es una tarea que se debe realizar cuidadosamente so pena de obtener resultados poco útiles. Es conveniente recordar que ningún criterio único puede dar una idea acabada del desempeño general de los *MOEA*s, ya que algunos se enfocan en la efectividad y otros en la eficiencia. También puede ser interesante tener una idea de la eficiencia y efectividad medidas en el tiempo, esto es, medir el progreso de un *MOEA* en cada generación; esto podría ser muy útil para establecer un criterio de parada que se ajuste a las necesidades. Todos estos aspectos pueden ser importantes al momento de juzgar a un *MOEA*. En las secciones siguientes se presentan las métricas escogidas para analizar los resultados de los experimentos presentados. La lista no debe considerarse

un completo compendio; más bien, sus componentes se han escogido de modo a tener un espectro de discusión lo suficientemente amplio y extendido.

Las métricas presentadas miden el desempeño en el dominio de los fenotipos (espacio objetivo). En otros trabajos [VEL99] se han presentado métricas que se enfocan en el genotipo (espacio de búsqueda). Como existe una correspondencia directa entre las soluciones Pareto óptimas y el frente Pareto óptimo, no se puede afirmar que un método es "mejor" que el otro. Sin embargo, es útil notar que diferentes soluciones podrían mapearse a un único vector objetivo.

Por otro lado, es conveniente recordar que como muchas métricas reflejan la similitud entre el frente Pareto óptimo real $\boldsymbol{Y}_{true}$ y el frente Pareto computado $\boldsymbol{Y}_{known}$ y como $\boldsymbol{Y}_{true}$ es en realidad nuestra incógnita, se ha construido $\boldsymbol{Y}_{true}$ uniendo todos los individuos no dominados hallados en todos los conjuntos solución calculados, de este modo, el frente Pareto óptimo real se aproxima mediante las mejores soluciones halladas en el curso de todos los experimentos.

El conjunto de *test* seleccionado para este trabajo comprende las métricas que se listan a continuación.

5.1.1. Generación de vectores no dominados (*GVND*)

Esta métrica cuenta el número de soluciones en el frente Pareto propuesto $\boldsymbol{Y}_{known}$. Se puede definir mediante la siguiente ecuación:

$$GVND \stackrel{\Delta}{=} |Y_{known}|_c \qquad \textbf{(5.1)}$$

donde $|\ |_c$ denota cardinalidad.

Schott [DUA01] usa esta métrica (aunque definida sobre el conjunto Pareto $\boldsymbol{X}_{known}$). Definir esta métrica en términos de genotipo o fenotipo es algo más bien subjetivo y tiene que ver con la propia preferencia; aunque conviene notar nuevamente que múltiples soluciones pueden mapearse a idénticos vectores objetivo. A pesar de que contar el número de soluciones no dominadas obtenidas ayuda a tener

una idea de la efectividad del *MOEA* para generar soluciones deseadas, la métrica no refleja que tan "lejos" se encuentran los vectores de $\boldsymbol{Y}_{known}$ de los de $\boldsymbol{Y}_{true}$.

5.1.2. Razón de generación de vectores no dominados (*RGVND*)

Como un valor muy bajo de *GVND* puede implicar poca efectividad, y un valor muy alto puede conducir a confusiones o mayor tarea para quien toma la decisión final, es difícil establecer un rango de valores deseables para *GVND*. Además, dependiendo del problema en cuestión y el modo en que se trata, la cardinalidad de $\boldsymbol{Y}_{known}$ puede variar considerablemente. Por tanto, resulta más útil indicar la razón entre la cantidad de vectores no dominados hallados y la cantidad de vectores no dominados existentes. A esta métrica conocemos como razón de generación de vectores no dominados (*RGVND*) y matemáticamente se define como:

$$RGVND \triangleq \frac{GVND}{|Y_{true}|_c} \qquad \textbf{(5.2)}$$

Como el objetivo es obtener un frente Pareto $\boldsymbol{Y}_{known}$ tan similar al verdadero frente Pareto como sea posible, un valor cercano a 1 es deseable.

5.1.3. Generación real de vectores no dominados (*GRVND*)

Aunque las métricas anteriores brinden buenos resultados, aún no dan idea clara del buen desempeño del algoritmo en la búsqueda, ya que el mismo puede dar como resultado un conjunto con igual cantidad de elementos que el frente Pareto óptimo real, pero sin que exista una correspondencia entre éstos y los elementos hallados. La métrica denominada generación real de vectores no dominados cuenta la cantidad de elementos en el frente Pareto hallado que en efecto pertenecen al frente Pareto óptimo real. Se define de la siguiente manera

$$GRVND \stackrel{\Delta}{=} \left| \left\{ y \mid y \in Y_{known} \quad \wedge \quad y \in Y_{true} \right\} \right|_c \tag{5.3}$$

5.1.4. Razón de Error (*E*)

Esta razón reporta la proporción de vectores objetivo en $\boldsymbol{Y}_{known}$ que no son miembros de $\boldsymbol{Y}_{true}$. Por ello una razón cercana a 1 indica una baja correspondencia entre el frente obtenido y el real; i.e. $E = 0$ es deseable. La definición matemática es:

$$E \stackrel{\Delta}{=} \frac{\sum_{i=1}^{N} e_i}{GVND}$$

$$e_i \begin{cases} 0 \text{ si un vector en } \mathrm{Y_{known}} \text{ esta también en } \mathrm{Y_{true}} \\ 1 \qquad\qquad\qquad\qquad\qquad \text{de otro modo} \end{cases} \tag{5.4}$$

Nótese que:

$$E = \frac{GVND - GRVND}{GVND} \tag{5.5}$$

por lo que en algunas publicaciones no se lo usa en conjunto con *GRVND* [VEL99].

5.1.5. Distancia generacional (*G*)

Presentada primeramente en [VEL99]; esta métrica es un valor que representa que tan lejos está $\boldsymbol{Y}_{known}$ de $\boldsymbol{Y}_{true}$. Se define como:

$$G \stackrel{\Delta}{=} \frac{\left(\sum_{i=1}^{N} d_i^2 \right)^{\frac{1}{2}}}{GVND} \tag{5.6}$$

donde d_i es la distancia euclidiana (en el espacio objetivo) entre cada vector objetivo $\boldsymbol{y} \in \boldsymbol{Y}_{known}$ y su miembro correspondiente más cercano en el frente Pareto óptimo real $\boldsymbol{Y}_{true}$. Un valor grande de G indica que $\boldsymbol{Y}_{known}$ está alejado de $\boldsymbol{Y}_{true}$; $G = 0$ es la situación ideal.

5.2. Descripción de los experimentos

Los siete algoritmos elegidos para este trabajo fueron implementados, para la resolución de los seis problemas de prueba presentadas en la Definición 2.9. Los algoritmos seleccionados se implementaron de acuerdo a la literatura original de referencia. En todos los casos los operadores de mutación y cruzamiento utilizados fueron mutación de un sólo bit y cruzamiento de un sólo punto.

Las soluciones de las distintas funciones de prueba se codificaron como cadenas binarias.

Todos los programas se escribieron en lenguaje C. Para las implementaciones paralelas se utilizaron las primitivas de comunicación proveídas por PVM v3.4.5 [PVM94]. Los programas se compilaron utilizando gcc -v2.96 para LINUX.

A fin de analizar el desempeño de las implementaciones paralelas de los MOEAs utilizados en este trabajo, para cada problema considerado se realizaron varias corridas distintas de los mismos. Las ejecuciones paralelas se realizaron utilizando una máquina PVM cuyas características principales se presentan en la Tabla 1. La asignación de procesos a cada CPU que compone el entorno paralelo utilizado lo realiza PVM.

Característica	Descripción
Tipo de computadora	COW
Tipo de CPU	AMD 800MHz
Memoria	256 MB
Sistema Operativo	Red Hat Linux v7.3
Red de comunicación	Ethernet 100 Mbps
Librería de comunicación	PVM 3.4.5

Tabla 1. Características del entorno computacional paralelo utilizado

Para las diferentes corridas, se han utilizado diferentes semillas para la generación de números aleatorios. Además, en una ejecución cada proceso pMOEA, utilizan semillas diferentes, explorando inicialmente en un subespacio mayor del espacio de búsqueda. En las distintas corridas paralelas se utilizó una probabilidad de migración (p_{mig}) igual a 0,5. En éstas, el número máximo de soluciones no dominadas que intercambian los distintos procesos (n_{mig}) es 10.

A continuación se listan los parámetros, según corresponda a cada MOEA utilizado:

- Tamaño de la población genética: 100.
- Tamaño de la población externa: 100.
- Probabilidad de cruzamiento: 0,8.
- Probabilidad de mutación: 0,01.
- Radio de nicho: 0,41
- Presión de dominancia: 10.
- Tasa de reducción: 0,7.

Por simplicidad, a las corridas que utilizan 1, 2, 4 y 8 procesadores se las llaman como corrida A, B, C y D respectivamente.

El número de procesos es de ocho, siendo uno el proceso Coordinador y los siete restantes procesos paralelos esclavos. El número de iteraciones para los pMOEA es de 1000 y para el proceso Coordinador es de 500.

Para el cálculo de las métricas que requieren el conocimiento del frente Pareto óptimo real, se utilizó como aproximación a $\boldsymbol{Y}_{true}$ el conjunto de soluciones no dominadas con respecto al conjunto unión de los resultados obtenidos considerando todas las ejecuciones realizadas.

5.3. Resultados de los experimentos sobre el problema ZDT1

En la Tabla 2 se aprecian los resultados de las métricas aplicadas (GRVND, GVND, RGVND, E y G), además del tiempo de cómputo en segundos requerido para llegar a las soluciones obtenidas sobre el problema ZDT1 con un procesador con los distintos conjuntos de algoritmos implementados. La Tabla 3 presenta los mismos resultados pero usando dos procesadores, la Tabla 4 utilizando cuatro procesadores y la Tabla 5 destinando ocho procesadores.

Conjunto	GRVND	GVND	RGVND	E	G	T(segundos)
CNSGA2	779	793	0,75667939	0,01765448	0,0003424	28003
MOGA	0	370	0,35305344	1	0,39598489	26578
NPGA	1	429	0,40935115	0,997669	0,28574904	26773
NSGA	3	449	0,42843511	0,99331849	0,05688096	26777
NSGA2	937	1132	1,08015267	0,17226148	0,00014616	28630
SPEA	801	978	0,93320611	0,1809816	0,00024582	28450
SPEA2	878	937	0,89408397	0,06296692	0,00024517	28532
TA-MOEA E	907	1009	0,96278626	0,10109019	0,0001414	29134
TA-MOEA P	896	925	0,88263359	0,03135135	0,00018936	29378
TA-MOEA S	810	953	0,90935115	0,15005247	0,00010155	28893

Tabla 2. Resultados sobre el problema ZDT1. Corridas Tipo A

Conjunto	GRVND	GVND	RGVND	E	G	T(segundos)
CNSGA2	781	794	0,75547098	0,0163728	0,00031443	21985
MOGA	0	369	0,3510942	1	0,39598124	20546
NPGA	1	429	0,40818268	0,997669	0,28574316	21001
NSGA	3	450	0,42816365	0,99333333	0,05688825	20921
NSGA2	938	1148	1,09229305	0,18292683	0,00017302	22003
SPEA	802	978	0,93054234	0,1799591	0,00026836	22567
SPEA2	876	943	0,89724072	0,07104984	0,00023492	22782
TA-MOEA E	909	1107	1,05328259	0,17886179	0,00011727	23460
TA-MOEA P	896	926	0,88106565	0,03239741	0,00011625	24128
TA-MOEA S	810	954	0,90770695	0,1509434	0,00011973	22961

Tabla 3. Resultados sobre el problema ZDT1. Corridas Tipo B

Conjunto	GRVND	GVND	RGVND	E	G	T(segundos)
CNSGA2	781	796	0,75665399	0,01884422	0,00039256	15151
MOGA	0	369	0,35076046	1	0,39592692	13921
NPGA	1	429	0,40779468	0,997669	0,28571671	14002
NSGA	3	450	0,42775665	0,99333333	0,05688654	13929
NSGA2	940	1150	1,09315589	0,1826087	0,00012431	14998
SPEA	802	980	0,93155894	0,18163265	0,00026429	15678
SPEA2	878	946	0,89923954	0,07188161	0,00029576	15978
TA-MOEA E	910	1101	1,04657795	0,17347866	0,00016616	17023
TA-MOEA P	897	928	0,88212928	0,03340517	0,0001863	16433
TA-MOEA S	811	955	0,90779468	0,15078534	0,00010197	16100

Tabla 4. Resultados sobre el problema ZDT1. Corridas Tipo C

Conjunto	GRVND	GVND	RGVND	E	G	T(segundos)
CNSGA2	782	799	0,75878443	0,0212766	0,0003441	7213
MOGA	0	370	0,35137702	1	0,39593433	6789
NPGA	1	429	0,40740741	0,997669	0,28573179	6987
NSGA	3	451	0,42830009	0,99334812	0,05683678	6890
NSGA2	946	1170	1,11111111	0,19145299	0,00013954	7513
SPEA	802	980	0,93067426	0,18163265	0,00025897	7811
SPEA2	878	946	0,89838557	0,07188161	0,00021274	7998
TA-MOEA E	911	1102	1,04653371	0,17332123	0,00010474	8003
TA-MOEA P	899	930	0,88319088	0,03333333	0,00014594	7988
TA-MOEA S	812	957	0,90883191	0,15151515	0,00018625	7987

Tabla 5. Resultados sobre el problema ZDT1. Corridas Tipo D

En la Tabla 6 se aprecian los resultados promedios de las métricas (GRVND, GVND, RGVND, E y G) aplicadas al problema ZDT1. Se puede apreciar que el conjunto NSGA2 es el mejor en la mayoría de las métricas consideradas. Con todo, puede apreciarse que los resultados experimentales usando TA-MOEA están muy próximos a los obtenidos con el NSGA2, superándolo en algunos casos.

Conjunto	GRVND	GVND	RGVND	E	G	T(segundos)
CNSGA2	780,75	795,5	0,75689819	0,0185418	0,00034837	18088
MOGA	0	369,5	0,35156993	1	0,39595684	16958,5
NPGA	1	429	0,40818268	0,997669	0,28573518	17190,75
NSGA	3	450	0,42816365	0,99333333	0,05687313	17129,25
NSGA2	940,25	1150	1,094196	0,1823913	0,00014576	18286
SPEA	801,75	979	0,93149382	0,18105209	0,00025936	18626,5
SPEA2	877,5	943	0,89724072	0,06945917	0,00024714	18822,5
TA-MOEA E	909,25	1079,75	1,0273549	0,15790692	0,00013239	19405
TA-MOEA P	897	927,25	0,882255	0,03262335	0,00015946	19481,75
TA-MOEA S	810,75	954,75	0,90842055	0,15082482	0,00012737	18985,25

Tabla 6. Resultados promedios sobre el problema ZDT1.

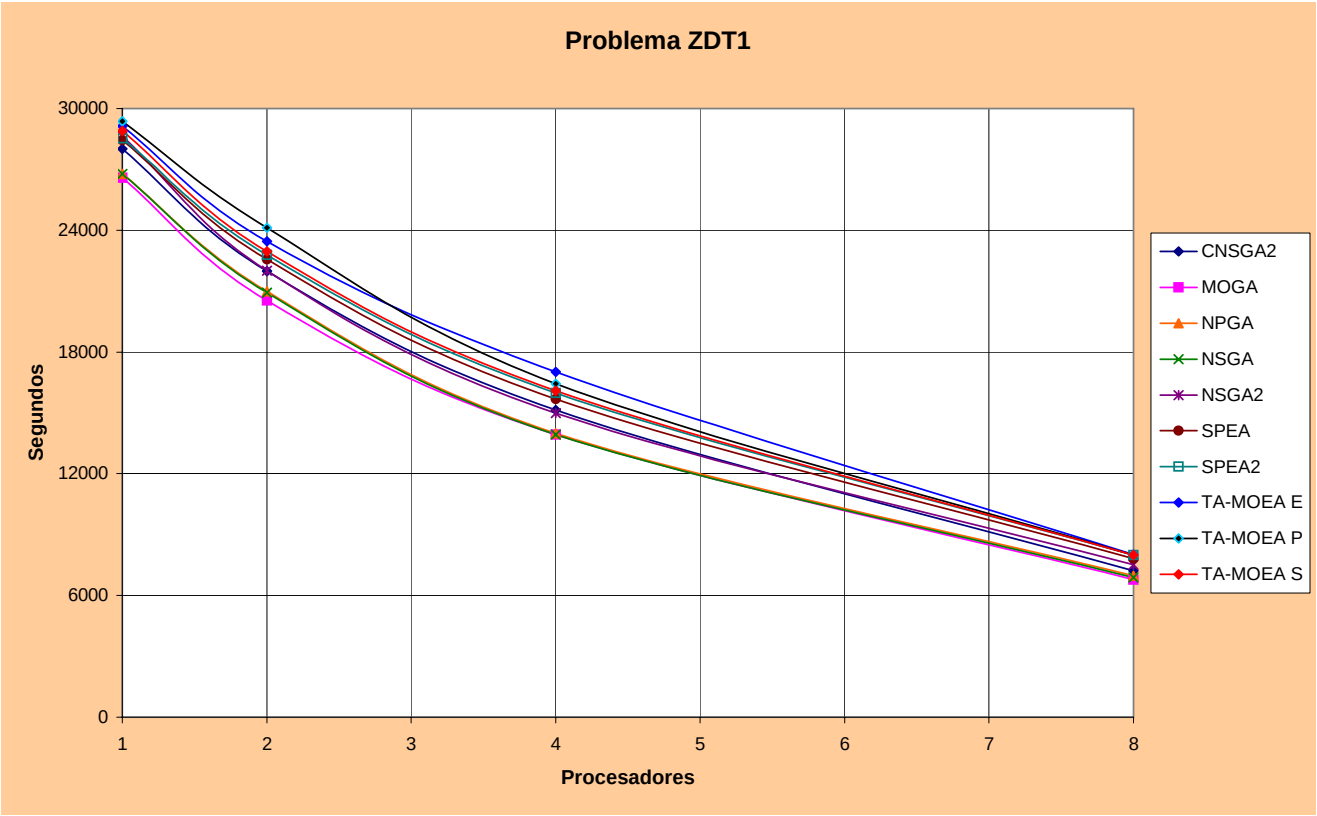

Figura 29. Tiempo Total de cada conjunto por número de Procesadores

En la Figura 29 se grafica el tiempo en segundos de cada conjunto para el problema ZDT1 en relación con el número de procesadores utilizado. En la figura Figura 30 se aprecia el tiempo total de cómputo requerido para los experimentos sobre el problema ZDT1.

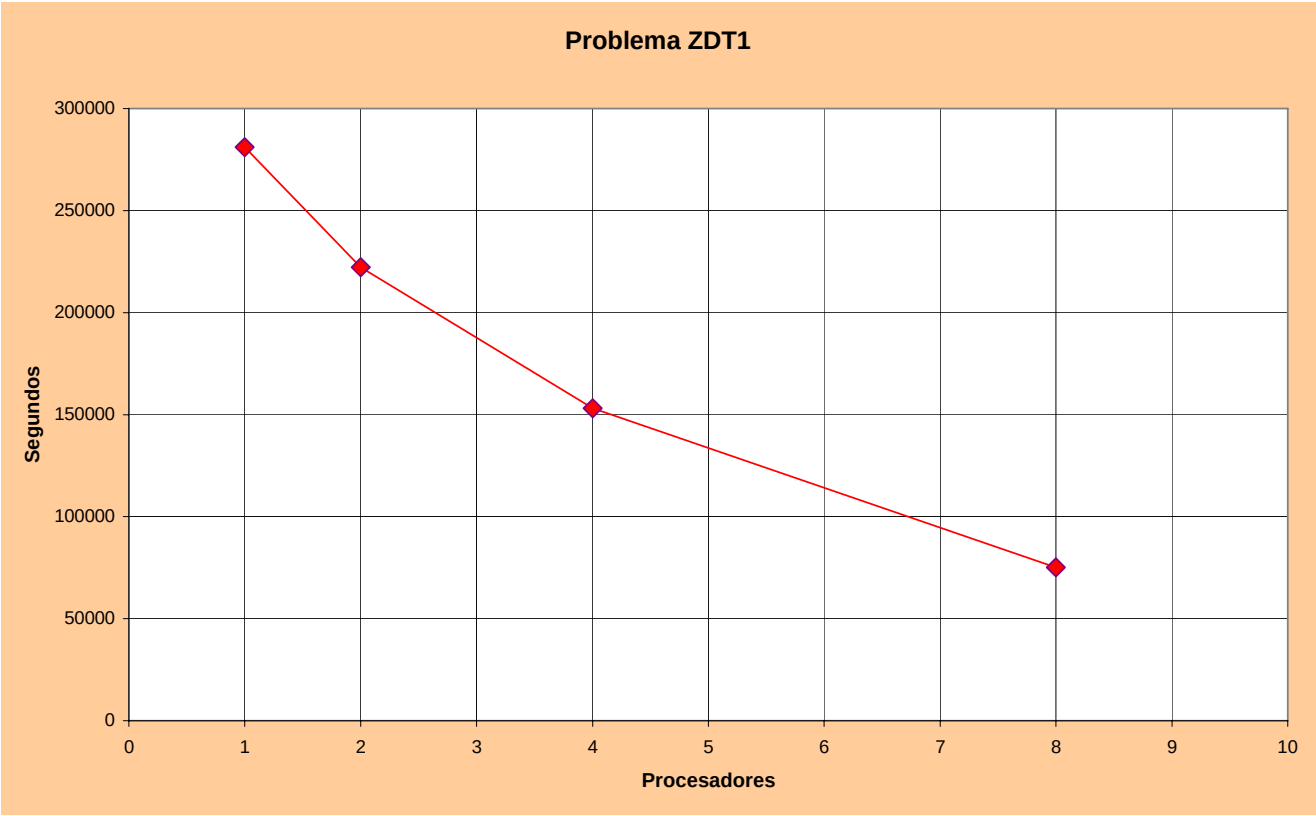

Figura 30. Tiempo Total de cada conjunto por número de Procesadores

5.4. Resultados de los experimentos sobre el problema ZDT2

En la Tabla 7 se aprecian los resultados de las métricas aplicadas (GRVND, GVND, RGVND, E y G), además del tiempo de cómputo en segundos requerido para llegar a las soluciones obtenidas sobre el problema ZDT2 con un procesador con los distintos conjuntos de algoritmos implementados. La Tabla 8 presenta los mismos resultados pero usando dos procesadores, la Tabla 9 utilizando cuatro procesadores y la Tabla 10 destinando ocho procesadores.

Conjunto	GRVND	GVND	RGVND	E	G	T(segundos)
CNSGA2	659	831	0,82768924	0,20697954	0,00026248	27192
MOGA	1	498	0,49601594	0,99799197	0,05626951	25846
NPGA	2	567	0,56474104	0,99647266	0,03053992	25324
NSGA	6	508	0,5059761	0,98818898	0,02504154	26493
NSGA2	820	902	0,89840637	0,09090909	0,00026521	28193
SPEA	866	924	0,92031873	0,06277056	0,00028308	29844
SPEA2	687	860	0,85657371	0,20116279	0,00033742	29123
TAMOEA E	818	901	0,89741036	0,09211987	0,00024268	29837
TAMOEA P	703	833	0,82968127	0,15606242	0,00025928	30007
TAMOEA S	712	834	0,83067729	0,14628297	0,00025644	29785

Tabla 7. Resultados sobre el problema ZDT2. Corridas Tipo A

Conjunto	GRVND	GVND	RGVND	E	G	T(segundos)
CNSGA2	659	831	0,82768924	0,20697954	0,00022	22390
MOGA	1	498	0,49601594	0,99799197	0,05628967	20890
NPGA	2	567	0,56474104	0,99647266	0,03055678	21033
NSGA	6	508	0,5059761	0,98818898	0,02506292	21030
NSGA2	820	902	0,89840637	0,09090909	0,00022397	22100
SPEA	866	924	0,92031873	0,06277056	0,00025478	23201
SPEA2	687	860	0,85657371	0,20116279	0,0003923	23120
TAMOEA E	818	901	0,89741036	0,09211987	0,00023385	23890
TAMOEA P	703	833	0,82968127	0,15606242	0,00026524	23001
TAMOEA S	712	834	0,83067729	0,14628297	0,00023941	23009

Tabla 8. Resultados sobre el problema ZDT2. Corridas Tipo B

Conjunto	GRVND	GVND	RGVND	E	G	T(segundos)
CNSGA2	659	831	0,82768924	0,20697954	0,00023784	18345
MOGA	1	498	0,49601594	0,99799197	0,05620197	16923
NPGA	2	567	0,56474104	0,99647266	0,03054215	17203
NSGA	6	508	0,5059761	0,98818898	0,02507486	17323
NSGA2	820	902	0,89840637	0,09090909	0,0002788	18239
SPEA	866	924	0,92031873	0,06277056	0,00029116	18923
SPEA2	687	860	0,85657371	0,20116279	0,00038378	19002
TAMOEA E	818	901	0,89741036	0,09211987	0,00023704	19856
TAMOEA P	703	833	0,82968127	0,15606242	0,00026184	19568
TAMOEA S	712	834	0,83067729	0,14628297	0,00027267	19534

Tabla 9. Resultados sobre el problema ZDT2. Corridas Tipo C

Conjunto	GRVND	GVND	RGVND	E	G	T(segundos)
CNSGA2	659	831	0,82768924	0,20697954	0,00023015	8739
MOGA	1	498	0,49601594	0,99799197	0,05620582	7344
NPGA	2	567	0,56474104	0,99647266	0,03056893	7791
NSGA	6	508	0,5059761	0,98818898	0,02506471	8001
NSGA2	820	902	0,89840637	0,09090909	0,00029733	8999
SPEA	866	924	0,92031873	0,06277056	0,00025516	9129
SPEA2	687	860	0,85657371	0,20116279	0,00034021	9186
TAMOEA E	818	901	0,89741036	0,09211987	0,00022048	9546
TAMOEA P	703	833	0,82968127	0,15606242	0,00026227	9523
TAMOEA S	712	834	0,83067729	0,14628297	0,00027287	9310

Tabla 10. Resultados sobre el problema ZDT2. Corridas Tipo D

En la Tabla 11 se aprecian los resultados promedios de las métricas (GRVND, GVND, RGVND, E y G) aplicadas al problema ZDT2. Se puede apreciar que el conjunto SPEA es el mejor en la mayoría de las métricas. Así mismo, se puede observar que los resultados experimentales usando TA-MOEAs están muy próximos a los obtenidos con el SPEA.

Conjunto	GRVND	GVND	RGVND	E	G	T(segundos)
CNSGA2	659	831	0,82768924	0,20697954	0,00023762	19166,5
MOGA	1	498	0,49601594	0,99799197	0,05624174	17750,75
NPGA	2	567	0,56474104	0,99647266	0,03055195	17837,75
NSGA	6	508	0,5059761	0,98818898	0,02506101	18211,75
NSGA2	820	902	0,89840637	0,09090909	0,00026633	19382,75
SPEA	866	924	0,92031873	0,06277056	0,00027105	20274,25
SPEA2	687	860	0,85657371	0,20116279	0,00036343	20107,75
TAMOEA E	818	901	0,89741036	0,09211987	0,00023351	20782,25
TAMOEA P	703	833	0,82968127	0,15606242	0,00026216	20524,75
TAMOEA S	712	834	0,83067729	0,14628297	0,00026035	20409,5

Tabla 11. Resultados promedios sobre el problema ZDT2.

5.5. Resultados de los experimentos sobre el problema ZDT3

En la Tabla 12 se aprecian los resultados de las métricas aplicadas (GRVND, GVND, RGVND, E y G), además del tiempo cómputo en segundos de requerido para llegar a las soluciones obtenidas sobre el problema ZDT3 utilizando un procesador con los distintos conjuntos de algoritmos implementados. La Tabla 13 presenta los mismos resultados pero usando dos procesadores, la Tabla 14 utilizando cuatro procesadores y la Tabla 15 destinando ocho procesadores.

Conjunto	GRVND	GVND	RGVND	E	G	T(segundos)
CNSGA2	588	753	0,90613718	0,21912351	0,01544332	34102
MOGA	6	500	0,60168472	0,988	0,14066316	32789
NPGA	33	675	0,81227437	0,95111111	0,1367797	33206
NSGA	58	547	0,65824308	0,89396709	0,01030922	33124
NSGA2	603	761	0,91576414	0,20762155	0,00169366	34000
SPEA	603	753	0,90613718	0,19920319	0,00799076	34230
SPEA2	788	805	0,96871239	0,02111801	0,00146	34001
TAMOEA E	702	801	0,96389892	0,12359551	0,00182585	34533
TAMOEA P	611	740	0,89049338	0,17432432	0,0009866	34523
TAMOEA S	609	742	0,89290012	0,17924528	0,00155131	34678

Tabla 12. Resultados sobre el problema ZDT3. Corridas Tipo A

Conjunto	GRVND	GVND	RGVND	E	G	T(segundos)
CNSGA2	589	754	0,90734055	0,21883289	0,01543631	28549
MOGA	7	500	0,60168472	0,986	0,14065475	27002
NPGA	34	678	0,81588448	0,94985251	0,13672665	26003
NSGA	58	550	0,66185319	0,89454545	0,01035954	26002
NSGA2	603	761	0,91576414	0,20762155	0,00160583	27931
SPEA	603	753	0,90613718	0,19920319	0,00795696	28483
SPEA2	788	805	0,96871239	0,02111801	0,00146968	28934
TAMOEA E	702	801	0,96389892	0,12359551	0,0018953	29634
TAMOEA P	606	738	0,88808664	0,17886179	0,00097455	29638
TAMOEA S	609	742	0,89290012	0,17924528	0,00157352	29374

Tabla 13. Resultados sobre el problema ZDT3. Corridas Tipo B

Conjunto	GRVND	GVND	RGVND	E	G	T(segundos)
CNSGA2	590	756	0,90865385	0,21957672	0,01541223	22321
MOGA	8	501	0,60216346	0,98403194	0,14066616	20893
NPGA	34	678	0,81490385	0,94985251	0,13673681	19288
NSGA	59	552	0,66346154	0,89311594	0,01031472	19283
NSGA2	603	761	0,91466346	0,20762155	0,00165654	22102
SPEA	603	753	0,90504808	0,19920319	0,00798166	22645
SPEA2	788	805	0,96754808	0,02111801	0,00146705	22675
TAMOEA E	702	801	0,96274038	0,12359551	0,00189895	23120
TAMOEA P	607	739	0,88822115	0,17861976	0,00099524	23003
TAMOEA S	609	742	0,89182692	0,17924528	0,00150677	22996

Tabla 14. Resultados sobre el problema ZDT3. Corridas Tipo C

Conjunto	GRVND	GVND	RGVND	E	G	T(segundos)
CNSGA2	590	756	0,90865385	0,21957672	0,01547122	15900
MOGA	8	501	0,60216346	0,98403194	0,14065111	13215
NPGA	34	678	0,81490385	0,94985251	0,13677709	13000
NSGA	60	554	0,66586538	0,89169675	0,01034896	13923
NSGA2	603	761	0,91466346	0,20762155	0,00161922	15203
SPEA	603	753	0,90504808	0,19920319	0,00796966	15923
SPEA2	788	805	0,96754808	0,02111801	0,00149731	16010
TAMOEA E	702	801	0,96274038	0,12359551	0,00188005	17923
TAMOEA P	611	740	0,88942308	0,17432432	0,00096995	17823
TAMOEA S	609	742	0,89182692	0,17924528	0,00154853	17902

Tabla 15. Resultados sobre el problema ZDT3. Corridas Tipo D

En la Tabla 16 se aprecian los resultados promedios de las métricas (GRVND, GVND, RGVND, E y G) aplicadas al problema ZDT3. Se puede apreciar que el conjunto SPEA2 es el mejor en la mayoría de las métricas. Además se puede observar que los resultados experimentales usando TA-MOEAs están muy próximos a los obtenidos con el SPEA2.

Conjunto	GRVND	GVND	RGVND	E	G	T(segundos)
CNSGA2	589,25	754,75	0,90769693	0,21927791	0,01544077	25218
MOGA	7,25	500,5	0,60192423	0,98551449	0,14065879	23474,75
NPGA	33,75	677,25	0,81449188	0,95016611	0,13675506	22874,25
NSGA	58,75	550,75	0,66235719	0,89332728	0,01033311	23083
NSGA2	603	761	0,91521347	0,20762155	0,00164381	24809
SPEA	603	753	0,9055923	0,19920319	0,00797476	25320,25
SPEA2	788	805	0,96812989	0,02111801	0,00147351	25405
TAMOEA E	702	801	0,9633193	0,12359551	0,00187504	26302,5
TAMOEA P	608,75	739,25	0,88905592	0,17653027	0,00098158	26246,75
TAMOEA S	609	742	0,8923632	0,17924528	0,00154503	26237,5

Tabla 16. Resultados promedios sobre el problema ZDT3.

5.5. Resultados de los experimentos sobre el problema ZDT4

En la Tabla 17 se aprecian los resultados de las métricas aplicadas (GRVND, GVND, RGVND, E y G), además del tiempo de cómputo en segundos requerido para llegar a las soluciones obtenidas sobre el problema ZDT4 con un procesador con los distintos conjuntos de algoritmos implementados. La Tabla 18 presenta los mismos resultados pero usando dos procesadores, la Tabla 19 utilizando cuatro procesadores y la Tabla 20 destinando ocho procesadores.

Conjunto	GRVND	GVND	RGVND	E	G	T(segundos)
CNSGA2	701	790	0,98997494	0,11265823	0,02099008	49210
MOGA	6	600	0,7518797	0,99	0,89585943	46302
NPGA	12	590	0,73934837	0,97966102	0,86593324	46987
NSGA	20	548	0,68671679	0,96350365	0,86574786	46703
NSGA2	601	780	0,97744361	0,22948718	0,34635952	48675
SPEA	699	834	1,04511278	0,1618705	0,69991682	50000
SPEA2	657	875	1,09649123	0,24914286	0,72848229	50097
TAMOEA E	690	892	1,11779449	0,2264574	0,0395947	50121
TAMOEA P	681	834	1,04511278	0,18345324	0,01605936	50098
TAMOEA S	673	753	0,94360902	0,1062417	0,01880382	51003

Tabla 17. Resultados sobre el problema ZDT4. Corridas Tipo A

Conjunto	GRVND	GVND	RGVND	E	G	T(segundos)
CNSGA2	701	791	0,99122807	0,11378003	0,02096666	38230
MOGA	6	601	0,75313283	0,99001664	0,89582069	37239
NPGA	12	591	0,7406015	0,97969543	0,86592776	37004
NSGA	20	548	0,68671679	0,96350365	0,86576237	37434
NSGA2	602	782	0,97994987	0,23017903	0,34635361	38342
SPEA	699	834	1,04511278	0,1618705	0,69990683	38273
SPEA2	657	875	1,09649123	0,24914286	0,72840972	38923
TAMOEA E	690	892	1,11779449	0,2264574	0,03952759	39756
TAMOEA P	681	834	1,04511278	0,18345324	0,01604109	39654
TAMOEA S	673	753	0,94360902	0,1062417	0,01884746	39216

Tabla 18. Resultados sobre el problema ZDT4. Corridas Tipo B

Conjunto	GRVND	GVND	RGVND	E	G	T(segundos)
CNSGA2	702	792	0,99123905	0,11363636	0,02093595	27700
MOGA	6	601	0,75219024	0,99001664	0,89584983	26674
NPGA	12	590	0,73842303	0,97966102	0,86594358	26334
NSGA	20	548	0,68585732	0,96350365	0,86575612	26324
NSGA2	602	783	0,97997497	0,2311622	0,34636143	28343
SPEA	699	834	1,04380476	0,1618705	0,69991211	28993
SPEA2	657	875	1,0951189	0,24914286	0,72848903	28386
TAMOEA E	690	892	1,11639549	0,2264574	0,03957495	28934
TAMOEA P	681	834	1,04380476	0,18345324	0,01607853	29055
TAMOEA S	673	753	0,94242804	0,1062417	0,01888087	29047

Tabla 19. Resultados sobre el problema ZDT4. Corridas Tipo C

Conjunto	GRVND	GVND	RGVND	E	G	T(segundos)
CNSGA2	702	792	0,99123905	0,11363636	0,0209728	19000
MOGA	6	602	0,7534418	0,99003322	0,89584238	17423
NPGA	12	592	0,74092616	0,97972973	0,86599521	17234
NSGA	20	548	0,68585732	0,96350365	0,86570806	17234
NSGA2	603	783	0,97997497	0,22988506	0,34635523	18234
SPEA	699	834	1,04380476	0,1618705	0,69992961	19439
SPEA2	657	875	1,0951189	0,24914286	0,72848508	19234
TAMOEA E	690	892	1,11639549	0,2264574	0,0395339	20374
TAMOEA P	681	834	1,04380476	0,18345324	0,01606766	203784
TAMOEA S	673	753	0,94242804	0,1062417	0,01880624	20343

Tabla 20. Resultados sobre el problema ZDT4. Corridas Tipo D

En la Tabla 21 se aprecian los resultados promedios de las métricas (GRVND, GVND, RGVND, E y G) aplicadas al problema ZDT4. Se puede apreciar que el conjunto cNSGA2 es el mejor de acuerdo a la mayoría de las métricas. Con todo, puede apreciarse que los resultados experimentales usando TA-MOEAs están muy próximos a los obtenidos con el cNSGA2, superándolo en algunos casos.

Conjunto	GRVND	GVND	RGVND	E	G	T(segundos)
CNSGA2	701,5	791,25	0,99092048	0,11342812	0,02096637	33535
MOGA	6	601	0,75266124	0,99001664	0,89584308	31909,5
NPGA	12	590,75	0,73982467	0,97968684	0,86594995	31889,75
NSGA	20	548	0,68628679	0,96350365	0,8657436	31923,75
NSGA2	602	782	0,97933626	0,23017903	0,34635745	33398,5
SPEA	699	834	1,04445836	0,1618705	0,69991634	34176,25
SPEA2	657	875	1,09580463	0,24914286	0,72846653	34160
TAMOEA E	690	892	1,11709455	0,2264574	0,03955778	34796,25
TAMOEA P	681	834	1,04445836	0,18345324	0,01606166	80647,75
TAMOEA S	673	753	0,94301816	0,1062417	0,01883459	34902,25

Tabla 21. Resultados promedios sobre el problema ZDT4.

5.7. Resultados de los experimentos sobre el problema ZDT5

En la Tabla 22 se aprecian los resultados de las métricas aplicadas (GRVND, GVND, RGVND, E y G), además del tiempo de cómputo en segundos requerido para llegar a las soluciones obtenidas sobre el problema ZDT5 con un procesador con los distintos conjuntos de algoritmos implementados. La Tabla 23 presenta los mismos resultados pero usando dos procesadores, la Tabla 24 utilizando cuatro procesadores y la Tabla 25 destinando ocho procesadores.

Conjunto	GRVND	GVND	RGVND	E	G	T(segundos)
CNSGA2	594	821	1,2127031	0,27649208	0,0032251	29345
MOGA	7	530	0,78286558	0,98679245	0,09825856	27886
NPGA	14	580	0,85672083	0,97586207	0,10005059	27349
NSGA	23	578	0,85376662	0,96020761	0,00904655	27343
NSGA2	583	853	1,25997046	0,31652989	0,00185406	29003
SPEA	583	850	1,25553914	0,31411765	0,03459335	29566
SPEA2	632	854	1,26144756	0,25995316	0,03013449	29434
TAMOEA E	587	830	1,22599705	0,29277108	0,03954038	30865
TAMOEA P	587	804	1,18759232	0,2699005	0,00293115	30677
TAMOEA S	586	824	1,21713442	0,28883495	0,01884133	30230

Tabla 22. Resultados sobre el problema ZDT5. Corridas Tipo A

Conjunto	GRVND	GVND	RGVND	E	G	T(segundos)
CNSGA2	594	821	1,21091445	0,27649208	0,00321941	25904
MOGA	7	533	0,78613569	0,98686679	0,09824908	23824
NPGA	14	580	0,85545723	0,97586207	0,10004111	23656
NSGA	23	578	0,85250737	0,96020761	0,00904643	24934
NSGA2	583	853	1,25811209	0,31652989	0,00186087	25875
SPEA	583	850	1,25368732	0,31411765	0,03450797	26003
SPEA2	632	854	1,25958702	0,25995316	0,0301318	26112
TAMOEA E	587	830	1,22418879	0,29277108	0,03956062	26932
TAMOEA P	584	801	1,18141593	0,27091136	0,00291815	26384
TAMOEA S	586	824	1,21533923	0,28883495	0,01886186	26044

Tabla 23. Resultados sobre el problema ZDT5. Corridas Tipo B

Conjunto	GRVND	GVND	RGVND	E	G	T(segundos)
CNSGA2	594	821	1,21091445	0,27649208	0,00329475	17548
MOGA	7	533	0,78613569	0,98686679	0,09821542	16234
NPGA	14	580	0,85545723	0,97586207	0,10008625	15332
NSGA	23	578	0,85250737	0,96020761	0,00907641	15834
NSGA2	583	853	1,25811209	0,31652989	0,00184453	17234
SPEA	584	851	1,25516224	0,31374853	0,03456182	17834
SPEA2	632	854	1,25958702	0,25995316	0,03019427	17902
TAMOEA E	587	830	1,22418879	0,29277108	0,03956372	18675
TAMOEA P	585	802	1,18289086	0,27057357	0,00292524	18773
TAMOEA S	586	824	1,21533923	0,28883495	0,01883557	18230

Tabla 24. Resultados sobre el problema ZDT5. Corridas Tipo C

Conjunto	GRVND	GVND	RGVND	E	G	T(segundos)
CNSGA2	594	821	1,20913108	0,27649208	0,00328059	10002
MOGA	7	533	0,78497791	0,98686679	0,09826079	9639
NPGA	14	580	0,85419735	0,97586207	0,10006975	9761
NSGA	23	578	0,85125184	0,96020761	0,00901004	9823
NSGA2	583	853	1,2562592	0,31652989	0,00184264	10665
SPEA	584	851	1,2533137	0,31374853	0,03453781	10783
SPEA2	632	854	1,25773196	0,25995316	0,03011728	10758
TAMOEA E	587	830	1,22238586	0,29277108	0,03958265	11234
TAMOEA P	586	802	1,18114875	0,26932668	0,00298319	11343
TAMOEA S	586	824	1,21354934	0,28883495	0,01884526	11332

Tabla 25. Resultados sobre el problema ZDT5. Corridas Tipo D

En la Tabla 26 se aprecian los resultados promedios de las métricas (GRVND, GVND, RGVND, E y G) aplicadas al problema ZDT5. Se puede apreciar que el conjunto SPEA2 es el mejor de acuerdo a la mayoría de las métricas. Además se puede observar que los resultados experimentales los TA-MOEAs están muy próximos a los obtenidos con el SPEA2.

Conjunto	GRVND	GVND	RGVND	E	G	T(segundos)
CNSGA2	594	821	1,21091445	0,27649208	0,00325496	20699,75
MOGA	7	532,25	0,7850295	0,98684829	0,09824596	19395,75
NPGA	14	580	0,85545723	0,97586207	0,10006193	19024,5
NSGA	23	578	0,85250737	0,96020761	0,00904485	19483,5
NSGA2	583	853	1,25811209	0,31652989	0,00185052	20694,25
SPEA	583,5	850,5	1,25442478	0,31393298	0,03455024	21046,5
SPEA2	632	854	1,25958702	0,25995316	0,03014446	21051,5
TAMOEA E	587	830	1,22418879	0,29277108	0,03956184	21926,5
TAMOEA P	585,5	802,25	1,18325959	0,27017763	0,00293943	21794,25
TAMOEA S	586	824	1,21533923	0,28883495	0,018846	21459

Tabla 26. Resultados promedios sobre el problema ZDT5

5.8. Resultados de los experimentos sobre el problema ZDT6

En la Tabla 27 se aprecian los resultados de las métricas aplicadas (GRVND, GVND, RGVND, E y G), además del tiempo de cómputo en segundos requerido para llegar a las soluciones obtenidas sobre el problema ZDT6 con un procesador con los distintos conjuntos de algoritmos implementados. La Tabla 28 presenta los mismos resultados pero usando dos procesadores, la Tabla 29 utilizando cuatro procesadores y la Tabla 30 destinando ocho procesadores.

Conjunto	GRVND	GVND	RGVND	E	G	T(segundos)
CNSGA2	430	662	1,02003082	0,35045317	0,0670369	43129
MOGA	23	402	0,61941448	0,94278607	0,20013065	39392
NPGA	28	580	0,89368259	0,95172414	0,10208448	40823
NSGA	23	578	0,89060092	0,96020761	0,09107543	40166
NSGA2	541	670	1,03235747	0,19253731	0,08069408	42658
SPEA	477	547	0,84283513	0,12797075	0,00215568	42839
SPEA2	459	690	1,06317411	0,33478261	0,00430239	42839
TAMOEA E	500	630	0,97072419	0,20634921	0,05955943	43890
TAMOEA P	487	523	0,80585516	0,06883365	0,01208099	43720
TAMOEA S	484	539	0,83050847	0,10204082	0,03489676	43621

Tabla 27. Resultados sobre el problema ZDT6. Corridas Tipo A

Conjunto	GRVND	GVND	RGVND	E	G	T(segundos)
CNSGA2	430	663	1,02157165	0,35143288	0,06708223	36433
MOGA	23	402	0,61941448	0,94278607	0,20011729	33239
NPGA	28	582	0,89676425	0,95189003	0,1020396	33923
NSGA	23	578	0,89060092	0,96020761	0,09105203	33834
NSGA2	541	670	1,03235747	0,19253731	0,08067138	36747
SPEA	477	547	0,84283513	0,12797075	0,00215678	36748
SPEA2	459	690	1,06317411	0,33478261	0,00434616	36784
TAMOEA E	501	630	0,97072419	0,2047619	0,05954464	37945
TAMOEA P	487	523	0,80585516	0,06883365	0,0120096	37458
TAMOEA S	485	540	0,83204931	0,10185185	0,03484446	37293

Tabla 28. Resultados sobre el problema ZDT6. Corridas Tipo B

Conjunto	GRVND	GVND	RGVND	E	G	T(segundos)
CNSGA2	430	663	1,02	0,35143288	0,06707196	28349
MOGA	23	402	0,61846154	0,94278607	0,2001286	25936
NPGA	28	583	0,89692308	0,95197256	0,10202666	26455
NSGA	23	578	0,88923077	0,96020761	0,09107043	26349
NSGA2	541	670	1,03076923	0,19253731	0,08067782	28374
SPEA	477	547	0,84153846	0,12797075	0,00219764	28647
SPEA2	459	690	1,06153846	0,33478261	0,00434739	28378
TAMOEA E	502	633	0,97384615	0,20695103	0,05958948	29371
TAMOEA P	487	523	0,80461538	0,06883365	0,0120452	29374
TAMOEA S	486	540	0,83076923	0,1	0,03487984	29120

Tabla 29. Resultados sobre el problema ZDT6. Corridas Tipo C

Conjunto	GRVND	GVND	RGVND	E	G	T(segundos)
CNSGA2	430	663	1,02	0,35143288	0,06702902	16729
MOGA	23	402	0,61846154	0,94278607	0,20013963	14863
NPGA	28	583	0,89692308	0,95197256	0,1020503	15004
NSGA	23	578	0,88923077	0,96020761	0,09107175	14977
NSGA2	541	670	1,03076923	0,19253731	0,08063101	16945
SPEA	477	547	0,84153846	0,12797075	0,00211199	16237
SPEA2	459	690	1,06153846	0,33478261	0,00434444	16889
TAMOEA E	502	633	0,97384615	0,20695103	0,05954674	16128
TAMOEA P	487	523	0,80461538	0,06883365	0,01200244	16009
TAMOEA S	486	540	0,83076923	0,1	0,03486608	15823

Tabla 30. Resultados sobre el problema ZDT6. Corridas Tipo D

En la Tabla 31 se aprecian los resultados promedios de las métricas (GRVND, GVND, RGVND, E y G) aplicadas al problema ZDT6. Se puede apreciar que el conjunto NSGA2 es el mejor de acuerdo a la mayoría de las métricas. Y los TA-MOEAS se acercan bastante a los obtenidos con el NSGA2.

Conjunto	GRVND	GVND	RGVND	E	G	T(segundos)
CNSGA2	430	662,75	1,02040031	0,35118823	0,06705503	31160
MOGA	23	402	0,61893764	0,94278607	0,20012904	28357,5
NPGA	28	582	0,8960739	0,95189003	0,10205026	29051,25
NSGA	23	578	0,88991532	0,96020761	0,09106741	28831,5
NSGA2	541	670	1,03156274	0,19253731	0,08066857	31181
SPEA	477	547	0,8421863	0,12797075	0,00215552	31117,75
SPEA2	459	690	1,06235566	0,33478261	0,0043351	31222,5
TAMOEA E	501,25	631,5	0,97228637	0,20625495	0,05956007	31833,5
TAMOEA P	487	523	0,8052348	0,06883365	0,01203456	31640,25
TAMOEA S	485,25	539,75	0,83102386	0,10097267	0,03487178	31464,25

Tabla 31. Resultados promedios sobre el problema ZDT6

5.9. Conclusiones Experimentales

Consideremos a continuación lo que ocurre con cada métrica al considerar simultáneamente los seis problemas ZDT con que se han realizado las corridas experimentales de este trabajo, de forma a intentar obtener una idea de cuál algoritmo es el más versátil para una gama completa de problemas diferentes.

Basados en los valores óptimos de las métricas, Tabla 32, se procede a clasificar los 10 conjuntos de algoritmos implementados para cada uno de los seis problemas ZDT con que se ha trabajado.

MÉTRICA	VALOR ÓPTIMO
Generación de vectores no dominados (GVND)	El mayor posible
Razón de generación de vectores no dominados (RGVND)	1
Generación real de vectores no dominados (GRVND)	El mayor posible
Razón de Error (E)	0
Distancia generacional (G)	0

Tabla 32. Valores óptimos de las métricas

En la Tabla 33 se puede notar que el TA-MOEA E se sitúa por delante de los demás siendo el mejor al considerar la métrica GRVND, por su parte los otros dos TA-MOEAs obtuvieron el quinto y sexto puesto.

	Conjunto	PROMEDIO	ZDT1	ZDT2	ZDT3	ZDT4	ZDT5	ZDT6
1	TAMOEA E	701,25	909,25	818	702	690	587	501,25
2	SPEA2	683,416667	877,5	687	788	657	632	459
3	NSGA2	681,541667	940,25	820	603	602	583	541
4	SPEA	671,708333	801,75	866	603	699	583,5	477
5	TAMOEA P	660,375	897	703	608,75	681	585,5	487
6	TAMOEA S	646	810,75	712	609	673	586	485,25
7	CNSGA2	625,75	780,75	659	589,25	701,5	594	430
8	NSGA	22,2916667	3	6	58,75	20	23	23
9	NPGA	15,125	1	2	33,75	12	14	28
10	MOGA	7,375	0	1	7,25	6	7	23

Tabla 33. Posiciones según la métrica GRVND

En la Tabla 34 se puede notar que el TA-MOEA E vuelve a obtener el primer lugar utilizando la métrica GVND, pero por su parte el TA-MOEA P obtuvo el quinto y el TA-MOEA S quedo en séptimo lugar.

	Conjunto	PROMEDIO	ZDT1	ZDT2	ZDT3	ZDT4	ZDT5	ZDT6
1	TAMOEA E	855,875	1079,75	901	801	892	830	631,5
2	NSGA2	853	1150	902	761	782	853	670
3	SPEA2	837,833333	943	860	805	875	854	690
4	SPEA	814,583333	979	924	753	834	850,5	547
5	TAMOEA P	776,458333	927,25	833	739,25	834	802,25	523
6	CNSGA2	776,041667	795,5	831	754,75	791,25	821	662,75
7	TAMOEA S	774,583333	954,75	834	742	753	824	539,75
8	NPGA	571	429	567	677,25	590,75	580	582
9	NSGA	535,458333	450	508	550,75	548	578	578
10	MOGA	483,875	369,5	498	500,5	601	532,25	402

Tabla 34. Posiciones según la métrica GVND

En la Tabla 35 sin embargo los TA-MOEAs no obtuvieron los primeros lugares usando la métrica RGVND, esto se debe a que el SPEA, SPEA2 y el NSGA2 dieron un número de soluciones mas próximos a la cantidad de elementos de $\boldsymbol{Y}_{true}$.

	Conjunto	PROXIMIDAD	ZDT1	ZDT2	ZDT3	ZDT4	ZDT5	ZDT6
1	SPEA	0,01692095	0,93149382	0,92031873	0,9055923	1,04445836	1,25442478	0,8421863
2	SPEA2	0,02328194	0,89724072	0,85657371	0,96812989	1,09580463	1,25958702	1,06235566
3	NSGA2	0,02947116	1,094196	0,89840637	0,91521347	0,97933626	1,25811209	1,03156274
4	TAMOEA E	0,03360905	1,0273549	0,89741036	0,9633193	1,11709455	1,22418879	0,97228637
5	CNSGA2	0,04758007	0,75689819	0,82768924	0,90769693	0,99092048	1,21091445	1,02040031
6	TAMOEA P	0,06100918	0,882255	0,82968127	0,88905592	1,04445836	1,18325959	0,8052348
7	TAMOEA S	0,06319295	0,90842055	0,83067729	0,8923632	0,94301816	1,21533923	0,83102386
8	NPGA	0,28687143	0,40818268	0,56474104	0,81449188	0,73982467	0,85545723	0,8960739
9	NSGA	0,32913226	0,42816365	0,5059761	0,66235719	0,68628679	0,85250737	0,88991532
10	MOGA	0,39897692	0,35156993	0,49601594	0,60192423	0,75266124	0,7850295	0,61893764

Tabla 35. Posiciones según la métrica RGVND

Tomando en cuenta la métrica E, el TA-MOEA P y el TA-MOEA S fueron los mejores como se aprecia en la Tabla 36. Por su parte el TA-MOEA E queda cuarto, ligeramente detrás del SPEA.

	Conjunto	PROMEDIO	ZDT1	ZDT2	ZDT3	ZDT4	ZDT5	ZDT6
1	TAMOEA P	0,14794676	0,03262335	0,15606242	0,17653027	0,18345324	0,27017763	0,06883365
2	TAMOEA S	0,16206707	0,15082482	0,14628297	0,17924528	0,1062417	0,28883495	0,10097267
3	SPEA	0,17446668	0,18105209	0,06277056	0,19920319	0,1618705	0,31393298	0,12797075
4	TAMOEA E	0,18318429	0,15790692	0,09211987	0,12359551	0,2264574	0,29277108	0,20625495
5	SPEA2	0,18926977	0,06945917	0,20116279	0,02111801	0,24914286	0,25995316	0,33478261
6	CNSGA2	0,19765128	0,0185418	0,20697954	0,21927791	0,11342812	0,27649208	0,35118823
7	NSGA2	0,20336136	0,1823913	0,09090909	0,20762155	0,23017903	0,31652989	0,19253731
8	NSGA	0,95979474	0,99333333	0,98818898	0,89332728	0,96350365	0,96020761	0,96020761
9	NPGA	0,97529112	0,997669	0,99647266	0,95016611	0,97968684	0,97586207	0,95189003
10	MOGA	0,98385957	1	0,99799197	0,98551449	0,99001664	0,98684829	0,94278607

Tabla 36. Posiciones según la métrica E

Los TAMOEA: TA-MOEA P y TA-MOEA S fueron los primeros también según la métrica G, esto se observa en la Tabla 37. Por su parte el TA-MOEA E queda cuarto, esta vez detrás del cNSGA2, mientras el SPEA queda ahora sexto.

	Conjunto	PROMEDIO	ZDT1	ZDT2	ZDT3	ZDT4	ZDT5	ZDT6
1	TAMOEA P	0,00540648	0,00015946	0,00026216	0,00098158	0,01606166	0,00293943	0,01203456
2	TAMOEA S	0,01241419	0,00012737	0,00026035	0,00154503	0,01883459	0,018846	0,03487178
3	CNSGA2	0,01788385	0,00034837	0,00023762	0,01544077	0,02096637	0,00325496	0,06705503
4	TAMOEA E	0,02348677	0,00013239	0,00023351	0,00187504	0,03955778	0,03956184	0,05956007
5	NSGA2	0,07182207	0,00014576	0,00026633	0,00164381	0,34635745	0,00185052	0,08066857
6	SPEA	0,12418788	0,00025936	0,00027105	0,00797476	0,69991634	0,03455024	0,00215552
7	SPEA2	0,12750503	0,00024714	0,00036343	0,00147351	0,72846653	0,03014446	0,0043351
8	NSGA	0,17635385	0,05687313	0,02506101	0,01033311	0,8657436	0,00904485	0,09106741
9	NPGA	0,25351739	0,28573518	0,03055195	0,13675506	0,86594995	0,10006193	0,10205026
10	MOGA	0,29784591	0,39595684	0,05624174	0,14065879	0,89584308	0,09824596	0,20012904

Tabla 37. Posiciones según la métrica G

Claramente al considerar el conjunto de problemas ZDT los TA-MOEAs presentan soluciones más robustas ante un conjunto de problemas con características y dificultades diferentes, convirtiéndose en una opción válida para la resolución de problemas de optimización multiobjetivo, especialmente en los casos en que no conocemos las características del problema a resolver, lo que ocurriría con problemas nuevos que intentamos resolver por primera vez.

En conclusión, existe una familia de algoritmos evolutivos multiobjetivo capaces de resolver problemas razonablemente complejos y de características muy diferentes (en cuanto a convexidad, continuidad, multimodalidad, etc.). Para cada tipo de problema, otro será el MOEA más adecuado para resolverlo, pero si consideramos un conjunto de problemas diferentes, los TA-MOEAs demostraron ser una excelente opción, dada su capacidad de resolver toda una familia de problemas diferentes con mejores promedios estadísticos que cada uno de los MOEAs corriendo individualmente.

Capítulo 6

Conclusiones y trabajos futuros

Como culminación del presente trabajo, en este capítulo se presentan las conclusiones finales y son propuestos algunos tópicos como posibles trabajos futuros.

6.1. Conclusiones Finales

En la primera parte del presente trabajo se propuso un modelo de TA-MOEA basado en un proceso Coordinador y varios procesos paralelos esclavos. Siete algoritmos evolutivos multiobjetivos fueron implementados en sus versiones paralelas, siguiendo el modelo propuesto, para la resolución de seis problemas de dificultad variada: los problemas de prueba ZDT1 a ZDT6.

Con estas implementaciones se llevaron a cabo varias corridas utilizando diferentes números de procesadores (1, 2, 4 y 8 procesadores).

Los resultados de estas corridas fueron comparados y analizados utilizando un conjunto de métricas de desempeño, habitualmente utilizados en trabajos similares.

A partir de los distintos resultados experimentales obtenidos, se han propuesto conclusiones parciales sobre la implementación de los TA-MOEAs en la resolución de cada uno de los problemas considerados. Con estos, es posible arribar a las siguientes conclusiones finales de este trabajo:

- El diseño e implementación de un equipo de algoritmos evolutivos multiobjetivo paralelos es un problema complejo. Existen varias decisiones que tomar. El rango de éstas va desde el tipo de plataforma paralela en que se realizará la implementación, hasta la determinación de diversos parámetros.
- Teniendo en cuenta las métricas utilizadas para medir la calidad del conjunto aproximación, se puede establecer que los TA-MOEA han sido los que lograron, en

general, un mejor desempeño promedio. Esta diferencia se extiende a toda la gama de problemas resueltos en este trabajo.

- Si bien los pMOEAs de primera generación obtienen pocas o ninguna solución, el tiempo de ejecución de los mismos es muy inferior pues están liberados de la utilización de mecanismos para la preservación de soluciones. Así, los procesos pMOEAs de segunda generación se encargan de proveer soluciones no dominadas a procesos pMOEAs de primera generación, con la esperanza que estos evolucionen hacia nuevas regiones del espacio de búsqueda encontrando soluciones no dominadas distintas. Este también es un mecanismo válido para promover la formación de diversidad lateral y preservación de información genética.
- Al utilizar TA-MOEAs, se extiende el espacio de búsqueda. La recepción de elementos provenientes de distintos algoritmos introduce información genética en forma aleatoria que es útil para la obtención de mejores soluciones.

En resumen a partir de los distintos resultados presentados queda claro que la utilización de TA-MOEAs es adecuada para la búsqueda de soluciones en espacios de búsqueda complejos y de alta dimensionalidad. Además se ha establecido la importancia de la utilización de implementaciones basados en elitismo y probabilidad para los TA-MOEAs.

De los conjuntos considerados en este trabajo se recomienda el uso del TA-MOEA E por quedar rankeado mejor, teniendo en cuenta todas las métricas utilizadas, y además por su capacidad de conocer razonablemente cual es el mejor MOEA para un determinado problema.

6.2. Trabajos Futuros

De forma a continuar con el trabajo iniciado en esta tesis, los siguientes tópicos son propuestos como trabajos futuros:

- modelado matemático de los algoritmos propuestos, de manera a hallar la complejidad de los mismos;
- mayor estudio sobre las diferentes alternativas de combinación de los algoritmos existentes.

- realizar un estudio en mayor profundidad de las distintas métricas y como se correlacionan.
- estudio e implementación de nuevos problemas de ingeniería que sirvan de base comparativa y experimental para los TA-MOEAs.
- comparar el desempeño de los distintos TA-MOEAs variando distintos parámetros, como ser las probabilidades de migración y número de soluciones migrantes.

Referencias

[BAC97] T. Bäck, D. B. Fogel, y Z. Michalewicz, editors. Handbook of Evolutionary Computation. Institute of Physics Publishing and Oxford University Press, 1997.

[BAR01] B. Barán, J. Vallejos, R. Ramos, y U. Fernández. Multi-objective reactive power compensation. En 2001 IEEE/PES Transmission and Distribution Conference and Exposition, volume 1, págs. 97–101. IEEE, 2001.

[BAR93] B. Barán. Estudio de Algoritmos Combinados Paralelos Asíncronos. Tesis Doctoral. Universidad Federal de Río de Janeiro – COPPE/UFRJ, Río de Janeiro - Brasil. Octubre de 1993.

[BAR95a] B. Barán, E. Kaszkurewicz y D. M. Falcão. Team Algorithms in Distributed Load Flow Computations. IEE Proceeding on Generation, Transmission and Distribution, Vol. 142, No. 6, pg. 583-588, noviembre 1995. Londres - Gran Bretaña.

[BAR95b] B. Barán, N. Cáceres y E. Chaparro. Reducción del Tiempo de Búsqueda utilizando una Combinación de Algoritmos Genéticos y Métodos Numéricos. XV International Conference of the Chilean Computer Science Society. Arica - Chile, 1995.

[BAR96] B. Barán, E. Kaszkurewicz y A. Bhaya. Parallel Asynchronous Team Algorithms: Convergence and Performance Analysis. IEEE Transactions on Parallel & Distributed Systems, Vol. 7, No. 7, pg. 677-688, julio 1996. Estados Unidos.

[BAR98] B. Barán, E. Chaparro y N. Cáceres. A-Teams en la Optimización del Caudal Turbinado de una Represa Hidroeléctrica. IBERAMIA-98, Lisboa-Portugal. 1998

[BAR99] B. Barán. Parallel Asynchronous Team Algorithms: Convergence and Perfomance Analysis. IEEE Transactions on Parallel & Distributed System, 7(7):677-688, Julio 1999.

[CAN98] E. Cantu-Paz. A survey of parallel genetic algorithms. Calculateurs Paralleles, Reseaux et Systemes Repartis, 10(2):141–171, 1998.

[CAN99a] E. Cantu-Paz. Designing efficient and accurate parallel genetic algorithms. Technical Report 2108, Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, Illinois, 1999.

[CAN99b] E. Cantú-Paz. A summary of research on parallel genetic algorithms. Technical Report 95007, Department of General Engineering, University of Illinois at Urbana-Champaign, Urbana, IL, 1999.

[CLE97] M. Clergue y P. Collard. Dual Genetic Algorithms and Pareto Optimization. En G. D. Smith, N. C. Steele, y R. F. Albrecht, editors, Artificial Neural Nets and Genetic Algorithms, págs. 188–197, Norwich, UK, Abril 1997. Springer-Verlag.

[COE01] C. A. Coello Coello y G. Toscano Pulido. A Micro-Genetic Algorithm for Multiobjective Optimization. En E. Zitzler, K. Deb, L. Thiele, C. A. Coello Coello, y D. Corne, editors, First International Conference on Evolutionary Multi-Criterion Optimization, págs. 126–140. Springer-Verlag. Lecture Notes in Computer Science No. 1993, 2001.

[COE02] C. A. Coello Coello y C. E. Mariano Romero. Evolutionary Algorithms and Multiple Objective Optimization. En M. Ehrgott y X. Gandibleux, editors, Multiple Criteria Optimization: State of the Art Annotated Bibliographic Surveys, págs. 277–331. Kluwer Academic Publishers, Boston, 2002.

[COE98] C. A. Coello Coello. An Updated Survey of GA-Based Multiobjective Optimization Techniques. Technical Report Lania-RD-98-08, Laboratorio Nacional de Informática Avanzada (LANIA), Xalapa, Veracruz, México, December 1998.

[COE99] C. A. Coello Coello. Constraint handling through a multiobjective optimization technique. En A. S. Wu, editor, Proceedings of the 1999 Genetic and Evolutionary Computation Conference. Workshop Program, págs. 117–118, Orlando, Florida, Julio 1999.

[COH78] J. Cohon, Multiobjective programming and planning, Academic Press, 1978.

[CRI04] J. Crichigno y B. Barán, “Multiobjective Multicast Routing Algorithm”. IEEE ICT’2004, Ceará, Brasil, 2004.

[DAR92] C. Darwin. El origen de las especies. Editorial Planeta-De Agostini, México, 1992.

[DEB00] K. Deb, S. Agrawal, A. Pratab, y T. Meyarivan. A Fast Elitist Non-Dominated Sorting Genetic Algorithm for Multi-Objective Optimization: NSGA-II. KanGAL report 200001, Indian Institute of Technology, Kanpur, India, 2000.

[DEB01] K. Deb y T. Goel. Controlled Elitist Non-dominated Sorting Genetic Algorithms for Better Convergence. En E. Zitzler, K. Deb, L. Thiele, C. A. Coello Coello, y D. Corne, editors, First International Conference on Evolutionary Multi-Criterion Optimization, págs. 67–81. Springer- Verlag. Lecture Notes in Computer Science No. 1993, 2001.

[DEB98] K. Deb. Multi-Objective Genetic Algorithms: Problem Difficulties and Construction of Test Problems, Technical Report CI-49/98, Dortmund: Department of Computer Science/LS11, University of Dortmund, Germany, 1998.

[DEB99] K. Deb. Evolutionary algorithms for multi-criterion optimization in engineering design. En K. Miettinen, M. M. Mäkelä, P. Neittaanmäki, y J. Periaux, editors, Evolutionary Algorithms in Engineering and Computer Science, págs. 135–161, Chichester, UK, 1999. JohnWiley & Sons, Ltd.

[DUA00] N. M. Duarte, A. E. Ruano, C. M. Fonseca, y P. J. Fleming. Accelerating Multi-Objective Control System Design Using a Neuro-Genetic Approach. En 2000 Congress on Evolutionary Computation, volume 1, págs. 392–397, Piscataway, New Jersey, Julio 2000. IEEE Service Center.

[DUA01] S. Duarte y B. Barán. Multiobjective Network Design Optimisation Using Parallel Evolutionary Algorithms. En XXVII Conferencia Latinoamericana de Informática CLEI-2001, Mérida, Venezuela, 2001.

[DUS71] Y. Dusonchet, S. Talukdar , H. Sinnot. Load Flows using a combination of Point Jacobi and Newton´s Method. IEEE Transactions on Power Apparatus and Systems, PAS-90, pp. 941 – 949, 1971.

[FLY95] R. Flynn y P. D. Sherman. Multicriteria Optimization of Aircraft Panels: Determining Viable Genetic Algorithm Configurations. International Journal of Intelligent Systems, 10:987–999, 1995.

[FON93] C. M. Fonseca y P. J. Fleming. Genetic Algorithms for Multiobjective Optimization: Formulation, Discussion and Generalization. En S. Forrest, editor, Proceedings of the Fifth International Conference on Genetic Algorithms, págs. 416–423, San Mateo, California, 1993. University of Illinois at Urbana-Champaign, Morgan Kauffman Publishers.

[FOS95] I. Foster. Designing and Building Parallel Programs. Addison-Wesley Publishing Company, Reading, MA, 1995.

[FOX81] S. Fox. An Organizational View of Distribuited Systems. IEEE Transactions on Systems, Man and Cybernetics, Vol. SMC-11, N° 1, pp. 70 – 80, 1981.

[GOL89] D. E. Goldberg. Genetic Algorithms in Search, Optimization and Machine Learning. Addison- Wesley Publishing Company, Reading, Massachusetts, 1989.

[HOR93] J. Horn y N. Nafpliotis. Multiobjective Optimization using the Niched Pareto Genetic Algorithm. Technical Report IlliGAl Report 93005, University of Illinois at Urbana-Champaign, Urbana, Illinois, USA, 1993.

[MAH95] S. W. Mahfoud. Niching methods for genetic algorithms. PhD thesis, University of Michigan, Urbana, IL, USA, 1995.

[MAR72] J. Marschak, R. Radner. Economic Theory of Teams. New Haven, CT: Yale University Press, 1972.

[MIC92] Z. Michalewicz. Genetic Algorithms + Data Structures = Evolution Programs. Springer-Verlag, 1992.

[MIE01] K. Miettien. Some methods for nonlinear multi-objetive optimization. En E. Zitzler, K. Deb, L. Thiele, C. A. Coello Coello, y D. Corne, editors, First International Conference on Evolutionary Multi-Criterion Optimization. Springer-Verlag. Lecture Notes in Computer Science No. 1993, 2001.

[PVM94] A. G. et al. PVM:Paralell Virtual machine - A user's guide and Tutorial for Networked parallel Computing. M.I.T. press, Cambridge, MA, 1994.

[RAM91] V. Ramesh, R. Quadrel, P. Souza, S. Talukdar, Asynchronous Teams: An Organizational Model for Distribuited Problem Solving.1991 Summer National Meeting, American Institute of Chemical Engineers, Pittsburgh, 1991.

[SRI93] N. Srinivas y K. Deb. Multiobjective optimization using nondominated sorting in genetic algorithms. Technical report, Department of Mechanical Engineering, Indian Institute of Technology, Kanpur, India, 1993.

[SRI94] N. Srinivas y K. Deb. Multiobjective Optimization Using Nondominated Sorting in Genetic Algorithms. Evolutionary Computation, 2(3):221–248, Fall 1994.

[SOU91] P. Souza, S. Talukdar. Genetic Algorithms in Asynchronous Teams. Proceedings of the Fourth International Conference on Genetic Algorithms (ICGA – 91), pp. 392 – 397, San Diego – California, 1991.

[STE86] R. Steuer, Multiple criteria optimization: theory, computation and application, New York, 1986.

[TAL82] S. Talukdar, S. Pyo, T. Giras. Asynchronous Procedures for Parallel Processing. Technical Report DRC – 18-54-82, Carnegie-Mellon University, Pittsburgh – Pennsylvania, 1982.

[TAL83] S. Talukdar, S. Pyo, R. Mehrotra. Designing Algorithms and Assigments for Distribuited Processing. Electric Power Research Institute. Final Report. Carnegie-Mellon University, Pittsburgh – Pennsylvania, 1983.

[TAL91] S. Talukdar, V. Ramesh, J. Nixon. A Distribuited System of Control Specialist for Real-Time Operations. Proceedings of the Third Symposium on Expert Systems Applications to Power Systems. Japan, 1991.

[VEL02] D. A. van Veldhuizen, J. B. Zydallis, y G. B. Lamont. Issues in Parallelizing Multiobjective Evolutionary Algorithms for Real World Applications. En Proceedings of the 17th ACM Symposium on Applied Computing, págs. 595–602, Madrid, Spain, 2002. ACM Press.

[VEL03] D. A. van Veldhuizen, J. B. Zydallis, y G. B. Lamont. Considerations in Engineering Parallel Multiobjective Evolutionary Algorithms. IEEE Transactions on Evolutionary Computation,7(2):144–173, April 2003.

[VEL99] D. A. van Veldhuizen. Multiobjective Evolutionary Algorithms: Classifications, Analyses, and New Innovations. PhD thesis, Department of Electrical and Computer Engineering. Graduate School of Engineering. Air Force Institute of Technology, Wright-Patterson AFB, Ohio, Mayo 1999.

[ZIT01] E. Zitzler, K. Deb, L. Thiele, C. A. Coello Coello, y D. Cornde, editors. Proceedings of the First International conference on EMOO, 2001, Berlin, Germany, Marzo 2001. Springer-Verlag.

[ZIT02] E. Zitzler, M. Laumanns, y L. Thiele. SPEA2: Improving the Strength Pareto Evolutionary Algorithm, in K. Giannakoglou, D. Tsahalis, J. Periaux, P. Papailou and T. Fogarty (eds.) EUROGEN 2001, Evolutionary Methods for Design, Optimization and Control with Applications to Industrial Problems, pp. 95--100, Athens, Greece, 2002.

[ZIT04] E. Zitzler, M. Laumanns y S. Bleuler. A Tutorial on Evolutionary Multiobjective Optimization, in Metaheuristics for Multiobjective Optimisation, pp. 3--37, Springer. Lecture Notes in Economics and Mathematical Systems Vol. 535, Berlin, 2004.

[ZIT98] E. Zitzler y L. Thiele. An Evolutionary Algorithm for Multiobjective Optimization: The Strength Pareto Approach. Technical Report 43, Computer Engineering and Communication Networks Lab (TIK), Swiss Federal Institute of Technology (ETH), Zurich, Switzerland, Mayo 1998.

[ZIT99] E. Zitzler, K. Deb, y L. Thiele. Comparison of Multiobjective Evolutionary Algorithms on Test Functions of Different Difficulty. En A. S. Wu, editor, Proceedings of the 1999 Genetic and Evolutionary Computation Conference. Workshop Program, págs. 121–122, Orlando, Florida, Julio 1999.