Un Nuevo Algoritmo Multiobjetivo Basado en Evolución Diferencial

Luis Vicente Santana Quintero y Carlos A. Coello Coello

CINVESTAV-IPN (Grupo de Computación Evolutiva)

Departamento de Ingeniería Eléctrica/Sección Computación

Av. IPN No. 2508, Col. San Pedro Zacatenco

México D.F. 07300, MÉXICO

lvspenny@hotmail.com, ccoello@cs.cinvestav.mx

Resumen

Este trabajo presenta un algoritmo multiobjetivo basado en la técnica de Evolución Diferencial. Éste se caracteriza por usar un fichero externo que utiliza dominancia-ϵ para mantener la diversidad del mismo, así como un doble mecanismo de selección que ayuda a que los hijos mejoren continuamente a sus padres. El nuevo algoritmo es validado adoptando funciones de prueba y métricas estándar de la literatura especializada. Los resultados obtenidos se comparan además con los producidos por otro esquema multiobjetivo basado en evolución diferencial (*Pareto Differential Evolution*), así como por dos algoritmos representativos del estado del arte en el área (el NSGA-II y el ϵ-MOEA).

1. Introducción

En este trabajo se atacan los problemas de optimización multiobjetivo. Estos problemas tienen más de un objetivo a optimizarse a la vez, y normalmente éstos se encuentran en conflicto unos con otros, por lo que el mejorar un objetivo implica empeorar el desempeño de los otros. Para resolver estos problemas multiobjetivo, en este trabajo se utilizó un tipo de algoritmo evolutivo llamado *Evolución Diferencial*, el cual fue desarrollado por Storn y Price en 1995 [8] para optimización en espacios continuos. Actualmente existen diversas técnicas para mantener las soluciones no dominadas obtenidas a lo largo del proceso evolutivo en un fichero externo. En este trabajo se utilizó el concepto de dominancia-ϵ que es una de las propuestas más recientes (y efectivas) en este sentido. La organización del resto del artículo es la siguiente. En la sección 2 se describe brevemente la Evolución Diferencial para optimización global. En la sección 3 se proporciona una revisión del trabajo previo relacionado con nuestra propuesta. Nuestra propuesta se describe en la sección 4. En la sección 5 se presenta la comparación de nuestra propuesta con respecto a otros algoritmos representativos del estado del arte en el área. después se revisan los trabajos previos que utilicen ED para optimización multiobjetivo, después se explica el algoritmo propuesto y se muestran algunos resultados obtenidos, por último las conclusiones en la Sección.

2. Evolución Diferencial

En la Evolución Diferencial (ED) [8], las variables se representan mediante números reales. La población inicial se genera de forma aleatoria y se seleccionan tres individuos como padres. Uno de ellos es el *padre principal* y éste se perturba con el vector de los otros dos padres. Si el valor resultante es mejor (cuando tiene un menor valor al evaluar la función objetivo) que el padre elegido, entonces lo reemplaza. De otra forma, se retiene al padre principal. Para cada vector $\overrightarrow{x_{i,G}}; i = 0, 1, 2, \ldots, N-1.$,

un vector de prueba $\overrightarrow{v}$ se genera de acuerdo a:

$$\overrightarrow{v} = \overrightarrow{x_{r1,G}} + F \cdot (\overrightarrow{x_{r2,G}} - \overrightarrow{x_{r3,G}})$$

con $r_1, r_2, r_3 \in [0, N-1]$, enteros y diferentes, y $F > 0$. Los enteros r_1, r_2 y r_3 son elegidos aleatoriamente en el intervalo [0, N-1] y son diferentes entre sí. F es un factor real y constante que controla la amplificación en la variación diferencial $(\overrightarrow{x_{r2,G}} - \overrightarrow{x_{r3,G}})$. Para decidir si el nuevo vector $\overrightarrow{v}$ se vuelve miembro de la población en la generación G+1, se compara con el vector $\overrightarrow{x_{i,G}}$, y si tiene un menor valor al evaluarla en la función objetivo entonces $\overrightarrow{v}$ se hace $\overrightarrow{x_{i,G+1}}$; de otra forma, el valor de $\overrightarrow{x_{i,G}}$ es retenido.

3. Trabajo Previo

Existen pocas propuestas para extender la Evolución Diferencial para resolver problemas multiobjetivo, algunas de ellas desarrolladas por Madavan en 2002 [6], Xue et al. en 2003 [11] y Babu & Jehan en 2003 [2]. A continuación se revisa la propuesta más representativa, la que también se utilizó para realizar comparaciones de desempeño con respecto a nuestro algoritmo.

3.1. PDE

El algoritmo llamado Pareto Differential Evolution (PDE) fue desarrollado por Abbass y Sarker en 2002 [1] y es una adaptación del algoritmo de ED descrito con anterioridad con las siguientes modificaciones:

1. Se genera una población inicial aleatoria con una distribución Gaussiana.
2. Todas las soluciones dominadas se quitan de la población y las restantes se utilizan para la reproducción.
3. Se realiza el cruce con la ED tomando a cada uno de los individuos de la población como padre principal y otros 3 padres se seleccionan aleatoriamente generando un hijo. Si el hijo generado domina al padre principal, entonces lo reemplaza.
4. Si el número máximo de soluciones no dominadas excede el máximo, entonces se aplica la función de distancia mínima D(x) (ver ec. (1)) para remover a los padres que estén más cerca y así preservar una buena distribución de soluciones finales.

$$D(x) = \frac{(min\|x - x^i\| + min\|x - x^j\|)}{2} \quad (1)$$

donde $x \neq x^i \neq x^j$.

4. Algoritmo Propuesto

Este algoritmo se basa en el cruce de la Evolución Diferencial para resolver problemas de optimización multiobjetivo, además de incluir el manejo de restricciones. En el algoritmo 1 se muestra el pseudo código del esquema que se propone.

```
- Generar una población inicial;
- Evaluar la aptitud de la población;
- FOR i = 0 hasta G
    REPETIR
        Seleccionar 3 padres;
        Realizar el cruce con ED;
        Realizar mutación uniforme;
        Evaluar aptitud del hijo;
        Si el HIJO es mejor que PADRE
            Reemplaza al PADRE;
    HASTA que se genera la siguiente
     población
    Marcar a los no dominados;
    Agregar los no dominados al fichero
     externo en caso de ser aceptados;
- Fin FOR
```

Algoritmo 1: Algoritmo Propuesto

4.1. Generación de la población inicial

La población inicial se genera de forma aleatoria con una distribución uniforme $U(0, 1)$, garantizando que las variables del problema se encuentren dentro de sus límites permisibles. Esto se realiza de la siguiente manera:

$$x_i = \mathrm{LI}_i + U(0, 1) \cdot (\mathrm{LS}_i - \mathrm{LI}_i)$$

donde: $i = 0, 1, 2, \ldots, v-1$. (v = número de variables de cada vector solución), LI_j y LS_j son los límites inferior y superior.

4.2. Selección

La selección se realiza de 2 maneras distintas, dependiendo del número de generaciones que hayan transcurrido. Este parámetro ejerce una presión de selección más alta ayudando a que se generen mejores hijos. Para controlar la presión de selección se adopta un parámetro sel_2 el cual puede tomar un valor desde 0.2 hasta 1 (pues cuando el proceso evolutivo inicia, el fichero externo está vacío). Este parámetro indica el número de generaciones que tienen que pasar hasta que se ejerza esta nueva presión de selección. Esto es:

$$\text{Selección} = \begin{cases} \text{Alea}, & \text{si } gen < (sel_2 * G_{max}); \\ \text{Elit}, & \text{de otra forma.} \end{cases}$$

donde gen es el número de generación actual y G_{max} es el número total de generaciones. Siempre se debe elegir un padre como referencia, el cual nos sirve para comparar el hijo generado por los tres padres con el primero. Los dos tipos de selección antes indicadas se explican a continuación:

1. **Selección Aleatoria.-** Se seleccionan 3 padres de la población primaria de forma aleatoria.
2. **Selección Elitista.-** Se seleccionan 3 padres de la población secundaria de forma aleatoria y además éstos deben cumplir con un factor de cercanía entre ellos $f_{cercania}$. Este factor está dado por:

   $$f_{cercania} = \frac{\sqrt{\sum_{i=0}^{FUN} (X_{i,max} - X_{i,min})^2}}{2^{FUN}}$$

   donde: FUN = número de funciones objetivo, $X_{i,max}$ = valor máximo de la i-ésima función objetivo del fichero externo, $X_{i,min}$ = valor mínimo de la i-ésima función objetivo del fichero externo.

4.3. Cruce utilizando la ED

En el algoritmo propuesto, se utiliza el esquema de la Evolución Diferencial detallado anteriormente (sección 2).

4.4. Mutación uniforme

Aunque la Evolución Diferencial no maneja el operador de mutación en ninguno de sus esquemas, en este trabajo se incluyó este operador debido a que en problemas con restricciones, el cruce por sí solo no lograba llegar a ciertas regiones del espacio de búsqueda. Dado que existe evidencia clara de que la mutación ayuda a la exploración de los algoritmos evolutivos, se eligió una mutación uniforme para nuestra propuesta.

4.5. Hijo vs padre

Una vez que se ha generado un hijo, se realiza la evaluación de éste con respecto a las funciones objetivo. Después se compara con el padre de referencia, y se elige a uno de ellos. Es importante mencionar que el esquema que se maneja para los problemas con restricciones requiere que éstas se normalicen. A continuación se darán a conocer las reglas de comparación entre el padre y el hijo para las funciones sin restricciones y después para aquellas que contienen restricciones.

4.5.1. Funciones sin restricciones

Primero se realiza una comparación de dominancia de Pareto entre padre e hijo:

* *uno domina al otro*, se le elige (padre o hijo)
* *si son no dominados entre ellos*, se realiza un $flip$ (0.5) para saber quién trasciende a la siguiente generación.

4.5.2. Funciones con restricciones

Para comparar individuos en problemas con restricciones, primero se verifica si son o no factibles, y después se checa la dominancia de Pareto, y se ve entonces lo siguiente:

* *si el padre es no factible y el hijo es no factible*, se elige al individuo que esté más cerca de la zona factible.

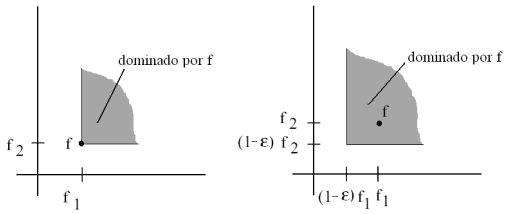

Figura 1: Concepto de dominancia (izq) y dominancia-ϵ (der)

* *si el padre es factible y el hijo es no factible*, se elige al hijo si y sólo si el hijo está a una distancia de 0.1 de la zona factible y además se hace un $flip$ (0.5). De lo contrario, se elige al padre.

* *si el padre es no factible y el hijo es factible*, se elige al padre si y sólo si el padre está a una distancia de 0.1 de la zona factible y además se hace un $flip$ (0.5). De lo contrario, se elige al hijo.

* *si el padre es factible y el hijo es factible*, se verifica la dominancia de Pareto y se elige tal y como si se tratara de un problema sin restricciones.

4.6. Concepto de dominancia-ϵ

El concepto de dominancia de Pareto se ilustra en la Figura 1 (izquierda) y se define de la siguiente manera: Si f, $g \in \Re^m$, entonces f se dice que domina a g, denotado como: $\vec{f} \prec \vec{g}$, si y sólo si:

$$1. \forall i \in \{1, \ldots, m\} : f_i \leq g_i$$

$$2. \exists j \in \{1, \ldots, m\} : f_j < g_j$$

De manera análoga, el concepto de dominancia-ϵ se ilustra en la Figura 1 (derecha) y se define de la siguiente manera. Si f, $g \in \Re^m$, entonces $\vec{f}$ se dice que ϵ-domina a $\vec{g}$ para algún $\epsilon > 0$, denotado como: $f \prec_\epsilon g$, si y sólo si para todo $i \in \{1, \ldots, \mathrm{m}\}$.

$$(1 - \epsilon) \cdot f_i \leq g_i$$

4.7. Mecanismo del fichero externo

Al nuevo individuo se le compara con cada uno de los miembros del fichero externo usando el criterio de dominancia-ϵ. A continuación se describe brevemente este procedimiento el cual se ha utilizado en otro algoritmo llamado $\epsilon - MOEA$ [4]. A cada solución dentro del fichero externo se le asigna un arreglo de identificación ($\mathbf{B} = (B_1, B_2, \ldots, B_M)^T$, donde M es el número total de funciones objetivo) como sigue:

$$B_j(f) = \begin{cases} (\lfloor (f_j - f_j^{min})/\epsilon_j \rfloor), & \text{para min } f_j; \\ (\lceil (f_j - f_j^{min})/\epsilon_j \rceil), & \text{para max } f_j. \end{cases}$$

donde: f_j^{min} es el valor mínimo posible del j-ésimo objetivo y ϵ_j es la tolerancia permitida en el j-ésimo objetivo. Ambos parámetros los debe definir el usuario [5]. Este arreglo de identificación divide todo el espacio de las funciones objetivo en hiper-cajas, cada una de las cuales tiene un tamaño ϵ_j en el j-ésimo objetivo.

1. Si el arreglo de identificación B_a de cualquier miembro del fichero externo a domina al nuevo individuo c_i, entonces este nuevo individuo es ϵ-dominado por un miembro del fichero externo y por lo tanto *no es aceptado*. Este caso se ilustra en la Figura 2 (a).
2. Si B_{c_i} del nuevo individuo domina a cualquier B_a del fichero externo, este miembro es eliminado y el nuevo individuo *es aceptado*. Este caso se ilustra en la Figura 2 (b).

   Si ninguno de los dos casos arriba mencionados ocurre, significa que el nuevo individuo es ϵ-no dominado con respecto a los miembros del fichero externo. Entonces surgen dos posibles casos:

   a) El nuevo individuo comparte el mismo vector $\mathbf{B}$ con algún individuo del fichero externo. Estos individuos se comparan con la dominancia convencional de Pareto y *se rechaza al dominado*. En caso de que ambos sean no dominados, se elige a la que esté mas cerca de la esquina inferior izquierda de la hiper-caja.

   b) Este caso se ilustra en la Figura 2 (d), y sucede cuando el nuevo individuo tiene una hiper-caja distinta a todos los

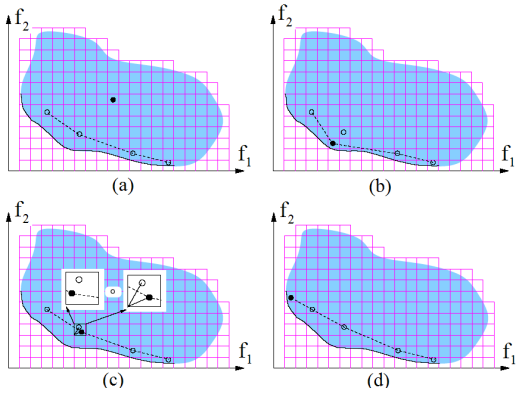

Figura 2: Concepto de dominancia-ϵ (para minimizar f_1 y f_2)

miembros del fichero externo, por lo que siempre *se acepta* este nuevo individuo.

5. Comparación de Resultados

Para validar nuestro algoritmo, se comparó contra PDE y contra otros dos algoritmos representativos del estado del arte en el área: el NSGA-II [3] y ϵ-MOEA [4]. Se realizaron 20 ejecuciones independientes para obtener los resultados de las tablas y se adoptaron 4 métricas de desempeño: Tasa de Error (TE) [10], Distancia Generacional (DG) [10], Distribución (D) [7] y Cobertura (C) [12]. Los parámetros utilizados para obtener los resultados para cada uno de los distintos algoritmos se muestran en la tabla 1. Se hace notar que se efectuaron 5,000 evaluaciones en los problemas sin restricciones y 25,000 en los problemas con restricciones. Nótese que en los algoritmos ϵ-MyDE y ϵ-MOEA, el tamaño del fichero externo depende del valor del vector-ϵ. El parámetro F es propio de la Evolución Diferencial.

Prob. 1 Este problema (ZDT1) fue propuesto en [12].

Minimizar:

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

Param	ϵ-My	NSGA-II	PDE	ϵ-MO
Población	100	100	100	100
Fich. Ext.	100	100	100	100
G_{max}	50	50	50	50
P_c	0.95	0.8	nd	1.0
P_m	1/N	1/N	nd	1/N
F	0.5	nd	N(0,1)	nd

donde: N = número de variables
nd = no requiere parámetro
G_{max} = 50 para problemas sin restricciones
G_{max} = 250 para problemas con restricciones

Tabla 1: Parámetros utilizados en los diversos algoritmos para realizar las pruebas y comparaciones.

donde:

$$\begin{aligned} f_1(\vec{x}) &= x_1, \\ f_2(\vec{x}) &= h(g(x_2,\ldots,x_m)f_1(x_1), g(x_2,\ldots,x_m)) \end{aligned}$$

con:

$$\begin{aligned} g(x_2,\ldots,x_m) &= 1 + 9 \cdot \sum_{i=2}^{m} \frac{x_i}{(m-1)} \\ h(f_1, g) &= 1 - \sqrt{(f_1/g)} \end{aligned}$$

y: $m = 30$; $x_i \in [0, 1]$. Las gráficas correspondientes al Prob. 1 se muestran en la Figura 3, y el resultado estadístico de la aplicación de las métricas se muestra en la tabla 2.

Gráficamente, se ve que ϵ-MyDE y ϵ-MOEA sí logran llegar al verdadero frente y con una buena distribución, aunque sin llegar a cubrirlo del todo. NSGA-II y PDE no logran llegar al verdadero frente, y su distribución no es buena.

Prob. 2 Este problema fue propuesto en [9]. Minimizar:

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

donde:

$$f_1(x, y) = x, f_2(x, y) = y$$

sujeto a:

$$\begin{aligned} 0 &\geq -(x^2) + -(y^2) + 1 + 0{,}1\cos(16\arctan\frac{x}{y}) \\ \frac{1}{2} &\geq (x - \frac{1}{2})^2 + (y - \frac{1}{2})^2 \end{aligned}$$

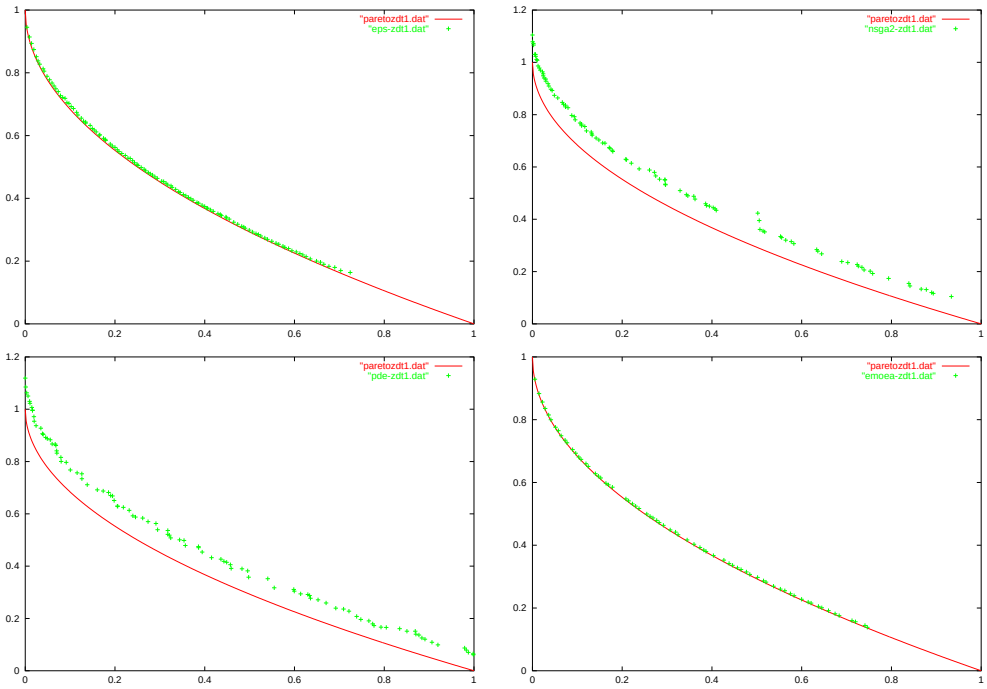

Figura 3: Frente de Pareto generado por ϵ-MyDE (arriba-izq), NSGA-II (arriba-der), PDE (abajo-izq) y ϵ-MOEA (abajo-der) para el problema 1.

Métrica-Algoritmo		media	mejor	peor	dev. Std
TE	ϵ-MyDE	**0.95**	**0.42**	**1**	0.1497
	NSGA-II	1	1	**1**	**0**
	PDE	0.9985	0.98	**1**	0.0048
	ϵ-MOEA	0.9581	0.8253	**1**	0.0529
DG	ϵ-MyDE	0.0018	**0.0001**	0.0213	0.0046
	NSGA-II	0.0089	0.0046	0.0167	0.0030
	PDE	707.1	0.0047	14142.1	3162.2
	ϵ-MOEA	**0.0004**	0.0002	**0.0007**	**0.0001**
D	ϵ-MyDE	0.0324	**0.0051**	0.2248	0.0495
	NSGA-II	0.0111	0.0056	0.0364	0.0067
	PDE	0.0099	0.0077	**0.0123**	**0.0013**
	ϵ-MOEA	**0.0083**	0.0065	0.0152	0.0019

C	ϵ-My	NS-II	PDE	ϵ-MO	Dominan
ϵ-MyDE		1	0.99	0.48	0.82
NSGA-II	0.005		0.78	0	0.26
PDE	1	1		1	1
ϵ-MOEA	0.87	0.91	0.95		**0.91**
Dominados	0.624	0.971	0.912	**0.494**	

Tabla 2: Resultados de Tasa de Error (TE), Distancia Generacional (DG), Distribución (D) y Cobertura (C) para el problema 1.

y: $-\pi \leq x$, $y \leq \pi$. Las gráficas correspondientes al Prob. 2 se muestran en la Figura 4, y el resultado estadístico de la aplicación de las métricas se muestra en la tabla 3.

PDE no logró resolver este problema, pues solo encontró un solo punto sobre el verdadero frente. Gráficamente, se ve que ϵ-MyDE cubre perfectamente todo el frente de Pareto verdadero y muestra una buena distribución. NSGA-II no logra cubrir todas las regiones del frente. ϵ-MOEA si alcanza las distintas regiones del frente, aunque no encontró muchas soluciones, ni mostró una buena distribución.

6. Conclusiones

En este trabajo se utilizó la evolución diferencial para resolver problemas multiobjetivo con y sin restricciones. Se pudo controlar la alta velocidad de convergencia que tiene la ED y se propuso un esquema que mantiene diversidad variando la forma de selección a cada cierto número de iteraciones. Se logró encontrar una técnica competitiva con respecto a otros algoritmos como NSGA-II, PDE y ϵ-MOEA. En cuanto al trabajo futuro, se puede mejorar el mecanismo para manejo de restricciones, para encontrar más soluciones dentro de la zona factible más rápido y explotar dichas soluciones dentro de la frontera de la factibilidad a fin de lograr mejores aproximaciones del verdadero frente de Pareto. En cuanto a la distribución de soluciones a lo largo del frente utilizando la dominancia-ϵ y su relación con el número de individuos no dominados, se podría encontrar una forma de auto-adaptar el valor de ϵ. Esto permitiría calcular de forma precisa el valor del vector-ϵ, sin perder la buena distribución ajustándose al número de soluciones requeridas. Actualmente se están realizando las pruebas y comparaciones con el nuevo algoritmo NSGA-II que se liberó en abril de 2005.

Agradecimientos

El primer autor agradece la beca otorgada por CONACyT para cursar estudios de posgrado en el CINVESTAV-IPN. El segundo autor agradece el apoyo del proyecto CONACyT No. 42435-Y.

Referencias

[1] Hussein A. Abbass and Ruhul Sarker. The Pareto Differential Evolution Algorithm. *International Journal on Artificial Intelligence Tools*, 11(4):531–552, 2002.

[2] B.V. Babu and M. Mathew Leenus Jehan. Differential Evolution for Multi-Objective Optimization. In *Proceedings of CEC'2003*, volume 4, pages 2696–2703, Canberra, Australia, December 2003. IEEE Press.

[3] Kalyanmoy Deb, Samir Agrawal, Amrit Pratab, and T. Meyarivan. A Fast Elitist Non-Dominated Sorting Genetic Algorithm for Multi-Objective Optimization: NSGA-II. In *Parallel Problem Solving from Nature VI*, pages 849–858, Paris, France, 2000.

[4] Kalyanmoy Deb, Manikanth Mohan, and Shikhar Mishra. Towards a Quick Computation of Well-Spread Pareto-Optimal Solutions. In *EMO 2003*, pages 222–236, Faro, Portugal, April 2003.

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

[6] Nateri K. Madavan. Multiobjective Optimization Using a Pareto Differential Evolution Approach. In *Proceedings of CEC'2002*, volume 2, pages 1145–1150, Piscataway, New Jersey, May 2002. IEEE.

[7] Jason R. Schott. Fault Tolerant Design Using Single and Multicriteria Genetic Algorithm Optimization. Master's thesis, Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, Cambridge, Massachusetts, May 1995.

[8] Rainer Storn and Kenneth Price. Differential evolution - a fast and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization*, 11:341–359, 1997.

[9] Masahiro Tanaka, Hikaru Watanabe, Yasuyuki Furukawa, and Tetsuzo Tanino. GA-Based Decision Support System for Multicriteria Optimization. In *Proceedings of the International Conference on Systems, Man, and Cybernetics*, volume 2, pages 1556–1561, Piscataway, NJ, 1995. IEEE.

[10] David A. Van Veldhuizen. *Multiobjective Evolutionary Algorithms: Classifications, Analyses, and New Innovations*. PhD thesis, Department of Electrical and Computer Engineering. Air Force Institute of Technology, USA, May 1999.

[11] Feng Xue, Arthur C. Sanderson, and Robert J. Graves. Pareto-based Multi-Objective Differential Evolution. In *Proceedings of CEC'2003*, volume 2, pages 862–869, Canberra, Australia, December 2003. IEEE.

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

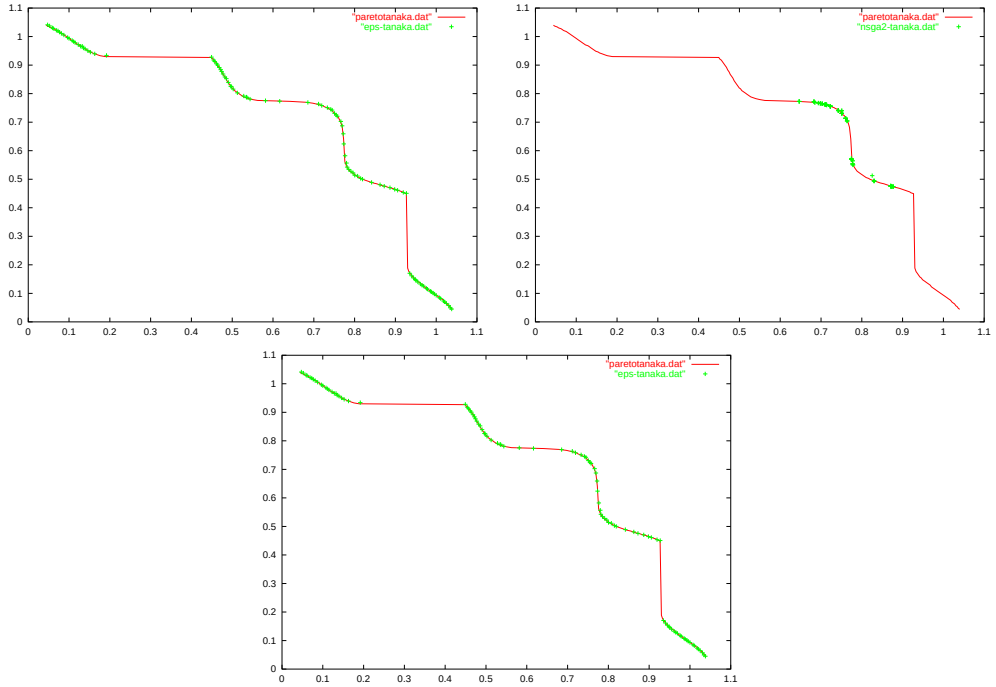

Figura 4: Frente de Pareto generado por ϵ-MyDE (arriba-izq), NSGA-II (arriba-der), y ϵ-MOEA (abajo) para el problema 2.

Métrica-Algoritmo		media	mejor	peor	dev Std
TE	e-MyDE	0.229	0.17	**0.323**	**0.0506**
	NSGA-II	**0.1**	**0**	0.34	0.089
	PDE	-	-	-	-
	ϵ-MOEA	0.319	0.127	0.468	0.078
DG	e-MyDE	**0.0004**	**0.0002**	**0.0006**	**8.5E-05**
	NSGA-II	0.0008	0.0003	0.0021	0.0005
	PDE	-	-	-	-
	ϵ-MOEA	0.0007	0.0004	0.0012	0.0001
D	e-MyDE	0.0074	0.0052	**0.0093**	**0.0010**
	NSGA-II	**0.005**	**0.0001**	0.018	0.0049
	PDE	-	-	-	-
	ϵ-MOEA	0.0136	0.0097	0.0176	0.0024

C	ϵ-My	NS-II	PDE	ϵ-MO	Dominan
e-MyDE		0.150	-	0.612	0.381
NSGA-II	0.564		-	0.658	**0.611**
PDE	-	-	-	-	-
ϵ-MOEA	0.410	0.126	-		0.268
Dominados	0.487	**0.138**	-	0.635	

Tabla 3: Resultados de Tasa de Error (TE), Distancia Generacional (DG), Distribución (D) y Cobertura (C) para el problema 2.