Una Propuesta de Adaptación en Línea para Optimización Evolutiva Multiobjetivo

Gregorio Toscano Pulido y Carlos A. Coello Coello

Resumen— **El ajuste (normalmente empírico) de parámetros es un problema común cuando se usan algoritmos evolutivos. En optimización evolutiva multiobjetivo esto se agudiza dada la complejidad de los problemas que suelen abordarse. En este artículo se propone un esquema de adaptación en línea que intenta lidiar con este problema, de tal forma que sea innecesario proporcionar parámetros a un algoritmo evolutivo multiobjetivo. Nuestra propuesta se aplica a un micro algoritmo genético que propusimos previamente para optimización multiobjetivo. En este trabajo proponemos también un esquema dinámico para elegir el operador de cruce más adecuado durante el proceso evolutivo. Dicho esquema ha contribuido a mejorar el desempeño de la nueva versión del algoritmo el cual es denominado micro-AG2 (μAG^2). La nueva técnica es comparada con respecto al NSGA-II, PAES y el micro-AG original usando diferentes funciones de prueba y métricas tomadas de la literatura especializada.**

Palabras clave— **optimización evolutiva multiobjetivo, algoritmos genéticos, adaptación en línea.**

I. Introducción

El uso de mecanismos de adaptación en línea y/o auto-adaptación es un tema que ha sido abordado de manera muy marginal en la literatura de optimización evolutiva multiobjetivo [1]. Sin embargo, los mecanismos de adaptación son sumamente importantes en optimización evolutiva multiobjetivo, pues su uso debido puede hacer innecesario el ajuste de parámetros que suelen requerir los algoritmos evolutivos. En nuestra investigación reciente hemos enfatizado la importancia de diseñar algoritmos evolutivos multiobjetivo que sean eficientes (computacionalmente hablando), y para ello propusimos un micro algoritmo genético (micro-AG) para optimización multiobjetivo. Este micro-AG usa una población de sólo cuatro individuos en un esquema en el que se adoptan tres formas distintas de elitismo y un proceso de reinicialización [2]. El micro-AG ha resultado altamente competitivo con respecto a otras técnicas evolutivas multiobjetivo que son representativas del estado del arte en el área (p.ej., el NSGA-II [3] y PAES [4]). Sin embargo, la principal desventaja del micro-AG es que requiere de varios parámetros (ocho en total) que deben calibrarse manualmente de una manera que puede no resultar intuitiva para el usuario.

Este artículo presenta una versión revisada del micro-AG para optimización multiobjetivo [2] en la cual no se requiere el ajuste manual de ningún parámetro por parte del usuario. La técnica resultante, llamada el micro-AG2 (μAG^2), tiene un desempeño promedio que resulta mejor que el micro-AG original, sin requerir el ajuste de ningún parámetro no intuitivo. Nuestro micro-AG2 es comparado con respecto a PAES [4], el NSGA-II [3] y nuestro micro-AG original [2] usando diferentes funciones de prueba (algunas de ellas con un alto grado de dificultad), que han sido propuestas recientemente en la literatura especializada.

CINVESTAV-IPN,
Depto. de Ing. Elect./Sección de Computación,
Av. IPN No. 2508, Col. San Pedro Zacatenco
México, D. F. 07300,
gtoscano@computacion.cs.cinvestav.mx,
ccoello@cs.cinvestav.mx

II. Conceptos Básicos

Definición 1 (POM General): *Encontrar el vector* $\vec{x}^* = [x_1^*, x_2^*, \ldots, x_n^*]^T$ *que satisfaga las* m *restricciones de desigualdad:*

$$g_i(\vec{x}) \geq 0 \quad i = 1, 2, \ldots, m \tag{1}$$

las p *restricciones de igualdad*

$$h_i(\vec{x}) = 0 \quad i = 1, 2, \ldots, p \tag{2}$$

y optimice el vector función

$$\vec{f}(\vec{x}) = [f_1(\vec{x}), f_2(\vec{x}), \ldots, f_k(\vec{x})]^T \tag{3}$$

donde $\vec{x} = [x_1, x_2, \ldots, x_n]^T$ *es el vector de las variables de decisión.* □

Definición 2 (Optimalidad de Pareto): *Un punto* $\vec{x}^* \in \Omega$ *(*Ω *es la región factible) es* **óptimo en el sentido de Pareto** *si para cada* $\vec{x} \in \Omega$ *y* $I = \{1, 2, \ldots, k\}$ *tanto,*

$$\forall_{i \in I} (f_i(\vec{x}) = f_i(\vec{x}^*)) \tag{4}$$

o, haya al menos un $i \in I$ *tal que*

$$f_i(\vec{x}) > f_i(\vec{x}^*) \tag{5}$$

□

En palabras, esta definición dice que $\vec{x}^*$ es un óptimo de Pareto si no existe vector $\vec{x}$ factible que haga decrementar algún criterio sin causar un aumento simultáneo en al menos algún otro. El significado de la frase "óptimo de Pareto" se considera con respecto a todo el espacio de las variables de decisión a menos que se indique lo contrario.

Definición 3 (Dominancia de Pareto): *Un vector $\vec{u} = (u_1, \ldots, u_k)$ se dice que domina a otro $\vec{v} = (v_1, \ldots, v_k)$ (denotado por $\vec{u} \preceq \vec{v}$) si y sólo si u es parcialmente menor que v, por ejemplo, $\forall i \in \{1, \ldots, k\},\ u_i \leq v_i \wedge \exists i \in \{1, \ldots, k\} : u_i < v_i$.* □

Definición 4 (Conjunto de óptimos de Pareto): *Dado un POM $\vec{f}(x)$, el conjunto de óptimos de Pareto ($\mathcal{P}^*$) se define como:*

$$\mathcal{P}^* := \{x \in \Omega \mid \neg\exists\, x' \in \Omega \ \ \vec{f}(x') \preceq \vec{f}(x)\}. \quad (6)$$

□

Definición 5 (Frente de Pareto): *Para un POM $\vec{f}(x)$ y conjunto de óptimos de Pareto $\mathcal{P}^*$ dado, el frente de Pareto ($\mathcal{PF}^*$) se define como:*

$$\mathcal{PF}^* := \{\vec{u} = \vec{f} = (f_1(x), \ldots, f_k(x)) \mid x \in \mathcal{P}^*\}. \quad (7)$$

□

III. Trabajo Relacionado

Han habido muy pocos intentos en la literatura para producir un algoritmo evolutivo multiobjetivo que adapte sus parámetros durante el proceso evolutivo y que, por tanto, no requiera del ajuste de ninguno de sus parámetros por parte del usuario. Una de las primeras propuestas para incorporar un mecanismo de auto-adaptación en un algoritmo evolutivo multiobjetivo es la debida a Kursawe [5]. En este trabajo se propone una estrategia evolutiva capaz de lidiar con problemas multiobjetivo en ambientes dinámicos. Laummans et al. [6] demostraron que una estrategia evolutiva auto-adaptativa estándar tiene problemas para converger al verdadero conjunto de Pareto de un problema de optimización multiobjetivo y propusieron un mecanismo alternativo de auto-adaptación que, sin embargo, fue aplicado únicamente en una función de aptitud agregativa. Tan et al. [7] propusieron el algoritmo evolutivo multiobjetivo incremental (IMOEA, por sus siglas en inglés), el cual usa un población cuyo tamaño varía en función de los tipos de soluciones producidas y de la densidad de la distribución de la población deseada. El IMOEA usa una medida de convergencia basada en la dominancia de población y en la tasa de progreso [8]. El IMOEA también usa nichos dinámicos (es decir, no se necesita definir el factor de compartición de aptitud).

El algoritmo propuesto en este artículo introduce un mecanismo de adaptación en línea en un algoritmo evolutivo multiobjetivo (el cual usa jerarquización de Pareto y una población secundaria), de tal manera que no se requiera ninguna calibración de sus parámetros. A diferencia de las propuestas previamente discutidas, nuestra técnica se centraliza principalmente en desempeñar una mejor exploración y explotación del espacio de búsqueda basándose en información muy simple y medidas estadísticas obtenidas del mismo proceso evolutivo.

IV. El μAG^2

Con la finalidad de hacer auto-contenido este artículo, procederemos a describir el funcionamiento del micro-AG para optimización multiobjetivo propuesto en [2]. Nuestro algoritmo primero genera una población de manera aleatoria. Dicha población se usa para alimentar la memoria de población, la cual se divide en dos partes: una porción reemplazable y otra no reemplazable. La porción no reemplazable nunca cambiará durante la ejecución del algoritmo y se utilizará como la fuente de diversidad del algoritmo. En contraste, la porción reemplazable experimentará cambios después de cada ciclo del micro-AG, con la finalidad de obtener una aproximación cada vez mejor del verdadero frente de Pareto.

La población del micro-AG es tomada, al inicio de cada uno de sus ciclos (con cierta probabilidad), de ambas porciones de la memoria de población. Por ende, se trabaja con una mezcla de individuos generados aleatoriamente (parte no reemplazable) con individuos que han sido objeto de un proceso evolutivo (parte reemplazable). Durante cada ciclo, el micro-AG utiliza operadores genéticos convencionales y opera con una población de sólo cuatro individuos. Una vez terminado un ciclo, se escogen dos vectores no dominados de la población final del micro-AG y se comparan con respecto a los "contendientes" de la memoria externa (obviamente, al inicio del proceso, la memoria externa estará vacía). Si alguno de estos individuos (o ambos) son no dominados con respecto al contenido de la memoria externa, entonces se les introduce a ésta. De tal forma que la memoria externa es nuestro fichero histórico de vectores no dominados encontrados a lo largo del proceso evolutivo. Todos los vectores contenidos en la memoria externa que resulten dominados

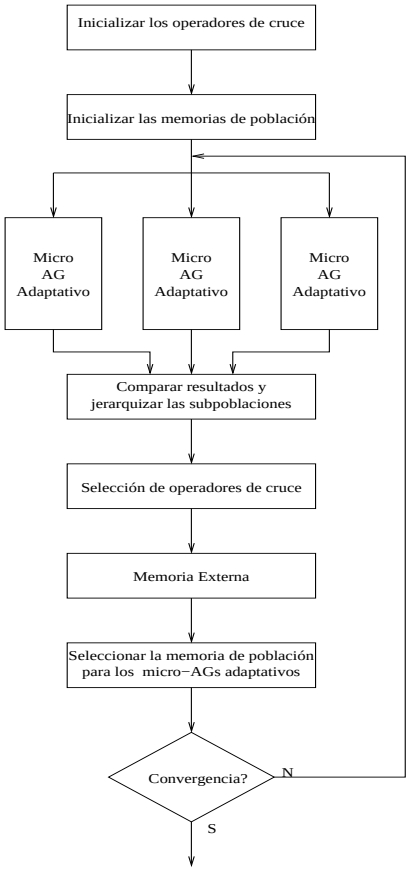

Fig. 1. Diagrama que ilustra la forma en la que trabaja nuestro μAG^2.

en esta comparación, son eliminados. Los mismos dos vectores antes mencionados se comparan también con respecto a dos elementos de la porción reemplazable de la memoria de población. Si alguno de estos vectores domina al elemento contra el que se le compara, entonces lo reemplaza. En caso contrario, el vector correspondiente es desechado y el elemento de la memoria reemplazable permanece inalterado. Con el tiempo, la parte reemplazable de la memoria de población tenderá a tener más vectores no dominados, algunos de los cuales serán usados en la población inicial del micro-AG.

La principal diferencia entre el μAG^2 y su antecesor es que la nueva versión cuenta con un mecanismo de adaptación en línea. La manera en la cual trabaja el μAG^2 se ilustra en la figura 1. Debido a que una de las principales características de la nueva técnica es el uso de una estrategia paralela para adaptar el operador de cruce (es decir, tenemos diferentes micro-AGs, los cuales son ejecutados en paralelo), comenzaremos por describir este mecanismo. Los tres operadores de cruce disponibles en nuestra técnica son: 1) SBX, 2) cruce de dos puntos y 3) un operador propuesto por nosotros. El comportamiento del operador de cruce aquí propuesto depende de la distancia de cada variable con respecto a sus padres: si las variables están más cerca de la varianza media de cada variable, entonces se efectúa una recombinación intermedia; de lo contrario, se efectúa una recombinación que enfatiza el surgimiento de soluciones alrededor de los padres. Estos tres operadores de cruce se seleccionaron debido a que presentaron el mejor desempeño de entre varios más que se evaluaron en una serie de experimentos que hemos realizado.

Una vez que los operadores de cruce han sido seleccionados, se generan aleatoriamente las memorias de población de los micro-AGs. Después, se ejecutan todos los micro-AGs internos, cada uno usando uno de los operadores de cruce disponibles (esto se realiza mediante un proceso determinista). Los vectores no dominados encontrados por cada micro-AG son comparados unos contra otros y se jerarquiza la contribución de cada operador de cruce con respecto a su efectividad para producir vectores no dominados. Posteriormente, se reemplaza el operador de cruce con peor desempeño por el que mostró mejor desempeño. La memoria externa almacena luego las soluciones no dominadas que se hayan obtenido

y se usan dichas soluciones para llenar la nueva memoria de población (de cada micro-AG interno).

Una vez completados todos estos procesos, se verifica si el algoritmo ha convergido. Para nuestros fines, consideramos que el algoritmo ha convergido si ninguno de los micro-AGs internos puede mejorar las soluciones alcanzadas previamente (más adelante se proporcionan más detalles al respecto). La lógica aquí es que si no se pueden encontrar nuevas soluciones en un cierto intervalo (razonablemente grande) de tiempo, es infructuoso seguir buscando.

El μAG^2 trabaja en dos etapas: la primera inicia con un proceso evolutivo convencional y concluye cuando la memoria externa de cada proceso esclavo está llena o cuando al menos un esclavo ha convergido (se presupone la misma definición de convergencia proporcionada anteriormente). La segunda etapa se concluye cuando se alcanza convergencia global (o sea, cuando todos los esclavos han convergido).

Las dos fases del μAG^2 se describen a continuación:

- **Fase de exploración:** En esta etapa, la mutación tiene más importancia que el cruce debido a que se necesita encontrar regiones promisorias del espacio de búsqueda. Por ende, se usa un porcentaje bajo de cruce y el operador de mutación es el reponsable principal de dirigir la búsqueda.

- **Fase de explotación:** En esta etapa, el operador de cruce tiene más importancia y por lo tanto la convergencia nominal se incrementa para obtener mejores resultados.

Debido a que el problema principal del micro-AG original es el que requiere varios parámetros adicionales que deben ser calibrados manualmente [2], el principal objetivo del μAG^2 fue precisamente la eliminación de dichos parámetros. Con este objetivo en mente, dividimos los parámetros del micro-AG en dos grupos: aquellos que no pueden ser adaptados en línea y aquellos que sí. La primera clase está compuesta por aquellos parámetros que dependen de las características específicas del problema e incluye lo siguiente: rangos de las variables de decisión, número de funciones objetivo y número de vectores no dominados que se aspira a encontrar (esto define el tamaño de la población secundaria al que llamaremos T_{pareto}).

Para aquellos parámetros que pueden ser adaptados, utilizamos un criterio basado en sus posibles dependencias. Ciertos parámetros tales como el porcentaje de mutación pueden ser fácilmente fijados mediante la división de uno entre el número de genes (o variables de decisión del problema), pero otros requieren de un análisis más profundo. Después de realizar dicho análisis, adoptamos la siguientes decisiones:

- **Porcentaje de cruce:** Es importante que se comporte de tal manera que se explore más al inicio del proceso evolutivo y que se explote más en etapas de búsqueda posteriores. De tal forma, durante la etapa de exploración, se usa únicamente un 50 % para el porcentaje de cruce, mientras que durante la etapa de explotación, se usa un 100 %.

- **Tamaño de la memoria de población:** El tamaño de la memoria de población fue fijado a $T_{pareto}/2$.

- **Porcentaje de memoria no reemplazable:** Debido a que el μAG^2 es un algoritmo paralelo, se decidió decrementar el porcentaje de memoria no reemplazable a un 10 % (con respecto a el 30 % usado en el micro-AG original [2]) del tamaño de la memoria de población.

- **Número total de iteraciones:** Esto se refiere al ciclo externo del micro-AG [2]. Este parámetro se fija de tal manera que se finalice después de que la memoria externa haya sido reemplazada $T_{pareto} \times 2$ veces sin tener individuos dominados o ningún reemplazo en los límites de la malla adaptativa.

- **Ciclo de reemplazo:** Reemplazamos la memoria reemplazable cuando T_{pareto} individuos han sido evaluados. Es importante hacer notar que cuando la memoria reemplazable es actualizada también se actualiza la memoria no reemplazable, mediante la generación aleatoria de individuos (como si fuese el inicio de la ejecución del algoritmo).

- **Número de subdivisiones de la malla adaptativa:** El número de subdivisiones se maneja con respecto al número de individuos deseados por región. Este valor se manipula de tal manera que nunca exceda (en promedio) tres individuos por región (de la malla) y nunca sea menor que 1.5 individuos por región. Por lo tanto el número de subdivisiones (y por ende el número de individuos por hipercubo) es incrementado o decrementado de acuerdo a estas reglas antes mencionadas.

El mecanismo de manejo de restricciones del micro-AG original se ha dejado intacto (ver [2] para detalles).

V. Funciones de Prueba y Resultados Numéricos

Para validar nuestro algoritmo tomamos varias funciones de prueba estándares que se encuentran en la literatura especializada. En todos esos casos, se generaron los verdaderos frentes de Pareto del problema usando un proceso de enumeración exhaustiva (con una cierta granularidad) de tal manera que pudiéramos realizar una comparación gráfica y otra basada en métricas de la calidad de las soluciones producidas por el μAG^2. También se compararon los resultados obtenidos por la técnica aquí propuesta contra los obtenidos por el Nondominated Sorting Genetic Algorithm II (NSGA-II) [3], la Pareto Archived Evolution Strategy (PAES) [4] y con respecto a nuestro micro-AG original [2]. Las métricas utilizadas para el estudio comparativo fueron: distancia generacional [9], tasa de error [8] y espaciado [10].

En los ejemplos siguientes, el NSGA-II usó una población de tamaño 100, un porcentaje de cruce de 0,8 (usando SBX), selección por torneo, y un porcentaje de mutación de $1/vars$, donde $vars$ =número de variables de decisión del problema. PAES fue ejecutado usando una profundidad de 5, un tamaño del fichero de 100, y un porcentaje de mutación de $1/bits$, donde bits se refiere a la longitud de la cadena del cromosoma que codifica las variables de decisión. El micro-AG original usó un porcentaje de cruce de 0,7, una memoria externa de 100 individuos, un valor de convergencia nominal de 2, una memoria de población de 50 individuos, un porcentaje de memoria no reemplazable de 0,3, una malla adaptativa, y un porcentaje de mutación de $1/L$ (L=longitud de la cadena cromosómica).

El número de evaluaciones de la función objetivo para el micro-AG original, el NSGA-II y PAES fue del valor más cercano al promedio de las evaluaciones obtenidas en 20 corridas por el μAG^2.

A. Función de prueba 1

Nuestro primer ejemplo es una función de prueba de n variables y n funciones objetivo propuesta recientemente por Deb et al. [11]:

$$\begin{array}{ll}
\text{Minimizar} & f_1(\mathbf{x}) = \mathbf{x_1}, \\
\vdots & \vdots \\
\text{Minimizar} & f_{M-1}(\mathbf{x}) = \mathbf{x_{M-1}}, \\
\text{Minimizar} & f_M(\mathbf{x}) = (\mathbf{1} + \mathbf{g}(\mathbf{x_M})) \\
 & h(f_1, f_2, \ldots, F_{M-1}, g), \\
\text{donde} & g(\mathbf{x_M}) = \mathbf{1} + \frac{\mathbf{9}}{|\mathbf{x_M}|} \sum_{\mathbf{x_i} \in \mathbf{x_M}} \mathbf{xi}, \\
 & h = M- \\
 & \sum_{i=1}^{M-1} \left[\frac{f_i}{1+g} (1 + \sin(3\pi f_i)) \right], \\
\text{sujeto a} & 0 \leq x_i \leq, \text{ for } i = 1, 2.., n,
\end{array} \tag{8}$$

Esta función de prueba tiene 2^{M-1} regiones Pareto óptimas desconectadas en el espacio de búsqueda. Deb et al. [11] propusieron usar 22 variables para hacerlo más desafiante y ese es precisamente el número de variables de decisión que fueron adoptadas para nuestro experimento. Los resultados para la primera función de prueba se reportan en la tabla I. Las figuras 2 y 3 muestran el comportamiento promedio de cada algoritmo con respecto a la distancia generacional. A partir de los resultados mostrados en la tabla I puede verse que el μAG^2 tuvo el mejor desempeño con respecto a la distancia generacional y al espaciado, y quedó en segundo lugar (después del NSGA-II) con respecto a la tasa de error.

B. Función de Prueba 2

El segundo ejemplo es un problema bi-objetivo propuesto por Deb [12]:

$$\begin{array}{ll}
\text{Minimizar} & f_1(x_1, x_2) = x_1 \\
\text{Minimizar} & f_2(x_1, x_2) = g(x_1, x_2) \cdot h(x_1, x_2)
\end{array} \tag{9}$$

donde:

$$g(x_1, x_2) = 11 + x_2^2 - 10 \cdot \cos(2\pi x_2) \tag{10}$$

$$h(x_1, x_2) = \begin{cases} 1 - \sqrt{\frac{f_1(x_1, x_2}{g(x_1, x_2}} & \text{si } f_1(x_1, x_2) \\ & \leq g(x_1, x_2) \\ 0 & \text{de lo contrario} \end{cases} \tag{11}$$

y $0 \leq x_1 \leq 1$, $-30 \leq x_2 \leq 30$.

Los resultados de la segunda función se resumen en la tabla II. Las figuras 4 y 5 muestran el comportamiento promedio de cada algoritmo con respecto a la distancia generacional. La tabla II muestra que el μAG^2 produjo los mejores resultados tanto para la distancia generacional como para la tasa de error. Con respecto al espaciado, quedó en segundo lugar, atrás del NSGA-II.

TABLA I

RESULTADOS OBTENIDOS EN LA PRIMERA FUNCIÓN DE PRUEBA (DTLZ6) POR EL μAG^2, EL NSGA-II, PAES Y EL MICRO-GA ORIGINAL.

		μ**AG²**		
Estadística	**Iteraciones**	**GD**	**ER**	**SP**
Promedio	20382	0.003561016	0.171	0.07382801
Mejor	16954	0.00304658	0.1	0.0598198
Peor	24394	0.00440405	0.25	0.0886338
Desv. Est.	2019.793840	0.000372	0.04290	0.007385
		PAES		
Promedio	20382	0.0161938745	0.49492855	0.125067925
Mejor	20382	0.00260934	0.2	0.0770419
Peor	20382	0.109795	0.75	0.258494
Desv. Est.	0	0.023217	0.1603101	0.049333
		NSGA-II		
Promedio	20100	0.003606146	0.115	0.077738055
Mejor	20100	0.00281355	0.07	0.039322
Peor	20100	0.0052915	0.16	0.0940669
Std.Desv.	0	0.000634	0.030174	0.012038
		micro-AG		
Promedio	20376	0.8760464	1.0425015	0.97022395
Mejor	20376	0.381188	1.025	0.232188
Peor	20376	1.66206	1.07143	3.4051
Desv. Est.	0	0.3524874	0.01302171	1.0298174

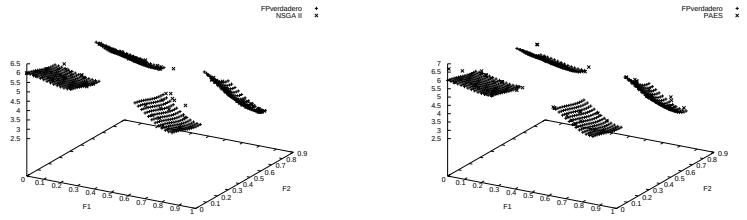

Fig. 2. Frentes de Pareto producidos por el NSGA-II (izquierda) y PAES (derecha) para la primera función de prueba.

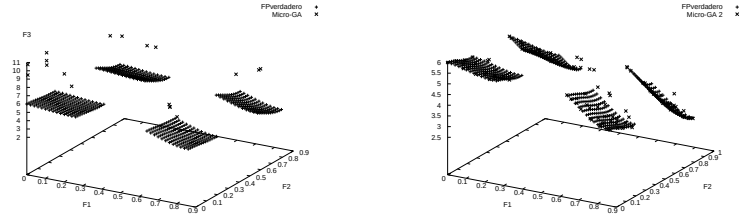

Fig. 3. Frentes de Pareto producidos por el micro-AG (izquierda) y el μAG^2 (derecha) en la primera función de prueba.

TABLA II

RESULTADOS OBTENIDOS EN LA SEGUNDA FUNCIÓN DE PRUEBA (DEB) POR EL μAG^2, EL NSGA-II, PAES Y EL MICRO-AG ORIGINAL.

		μ**AG²**		
Estadística	**Iteraciones**	**GD**	**ER**	**SP**
Promedio	9171.8	0.00016127085	0.115	0.0088751215
Mejor	6186	0.000102157	0	0.00721023
Peor	13826	0.000218467	0.32	0.0100066
Desv. Est.	0.081917	1956.912487	4.252223-05	0.000822
		PAES		
Promedio	9171	0.4651514	0.70408545	5.46232964
Mejor	9171	0.242424	0.0252054	0.0829736
Peor	9171	1	7.97044	64.8108
Desv. Est.	0	0.180424	2.012568	16.406210
		NSGA-II		
Promedio	9100	0.0002118179	0.2105	0.0079981215
Mejor	9100	0.000155758	0.01	0.00646298
Peor	9100	0.000282185	0.74	0.0089998
Desv. Est.	0	3.577123-05	0.224252	0.000594
		micro-AG		
Promedio	91068	0.0556739552	0.162	0.281928729
Mejor	91068	0.000159071	0.05	0.00637886
Peor	91068	0.465348	0.31	1.22778
Desv. Est.	0	0.1079727	0.0796439	0.3647516

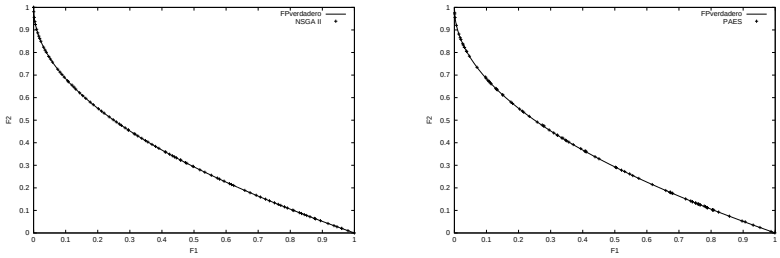

Fig. 4. Frentes de Pareto producidos por el NSGA-II (izquierda) y PAES (derecha) en la segunda función de prueba.

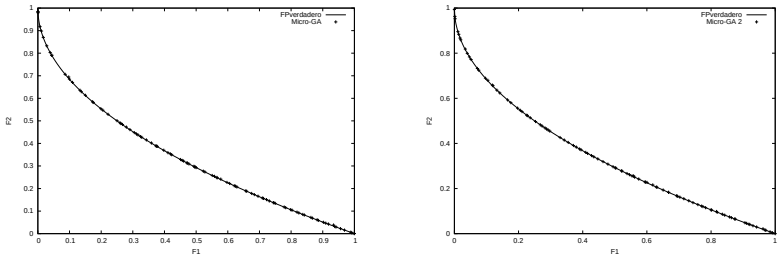

Fig. 5. Frentes de Pareto producidos por el micro-AG (izquierda) y el μAG^2 (derecha) en la segunda función de prueba.

VI. Conclusiones y Trabajo Futuro

En este trabajo se ha propuesto un esquema de adaptación en línea que permite el uso del micro-AG para optimización multiobjetivo sin necesidad de definir *a priori* ningún parámetro no intuitivo. El enfoque propuesto obtiene información del proceso evolutivo para guiar la búsqueda eficientemente. Entre otras cosas, nuestro esquema decide cuál es el operador de cruce más apropiado y cambia entre una etapa de exploración y una de explotación manipulando la importancia del cruce y de la mutación. También se ha definido un criterio que permite detener la ejecución del algoritmo cuando la búsqueda parece infructuosa y, por tanto, no se requiere definir un número máximo de generaciones durante las cuales se ejecutará el algoritmo.

El enfoque propuesto ha sido validado con varias funciones de prueba, de las cuales se incluyeron dos en este artículo. Nuestros resultados se compararon con respecto a nuestro micro-AG original y con respecto a PAES y al NSGA-II, usando tres métricas: distancia generacional, tasa de error y espaciado. Nuestros resultados preliminares indican que aunque nuestra técnica no siempre derrota a los demás algoritmos, se mantiene como muy competitiva y normalmente mejora los resultados producidos por el micro-AG original sin la necesidad de requerir ningún parámetro definido manualmente por el usuario.

Como parte de nuestro trabajo futuro, queremos experimentar con estructuras de datos espaciales (p.ej., quadtrees [13]) para eficientar el almacenamiento y la recuperación de vectores no dominados de la población secundaria. También estamos trabajando en un mecanismo que reduzca el número de evaluaciones de la función de aptitud que se requieren para aproximar razonablemente bien el frente de Pareto de un problema. Así mismo, nos interesa experimentar con mecanismos para incorporar preferencias del usuario [14].

Agradecimientos

El primer autor agradece el apoyo brindado por el Consejo Nacional de Ciencia y Tecnología (CONACyT) a través de una beca para cursar estudios de posgrado en la sección de Computación del departamento de Ingeniería Eléctrica del CINVESTAV-IPN. El segundo autor agradece el apoyo brindado por CONACyT a través del proyecto 34201-A.

Referencias

[1] Carlos A. Coello Coello, David A. Van Veldhuizen, and Gary B. Lamont, *Evolutionary Algorithms for Solving Multi-Objective Problems*, Kluwer Academic Publishers, New York, May 2002, ISBN 0-3064-6762-3.

[2] Carlos A. Coello Coello and Gregorio Toscano Pulido, "Multiobjective Optimization using a Micro-Genetic Algorithm," in *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'2001)*, Lee Spector et al., Ed., San Francisco, California, 2001, pp. 274–282, Morgan Kaufmann Publishers.

[3] Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and T. Meyarivan, "A Fast and Elitist Multiobjective Genetic Algorithm: NSGA–II," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, April 2002.

[4] Joshua D. Knowles and David W. Corne, "Approximating the Nondominated Front Using the Pareto Archived Evolution Strategy," *Evolutionary Computation*, vol. 8, no. 2, pp. 149–172, 2000.

[5] Frank Kursawe, "A Variant of Evolution Strategies for Vector Optimization," in *Parallel Problem Solving from Nature. 1st Workshop, PPSN I*, H. P. Schwefel and R. Männer, Eds., Berlin, Germany, oct 1991, vol. 496 of *Lecture Notes in Computer Science*, pp. 193–197, Springer-Verlag.

[6] Marco Laumanns, Günter Rudolph, and Hans-Paul Schwefel, "Mutation Control and Convergence in Evolutionary Multi-Objective Optimization," in *Proceedings of the 7th International Mendel Conference on Soft Computing (MENDEL 2001)*, Brno, Czech Republic, June 2001.

[7] K.C. Tan, T.H. Lee, and E.F. Khor, "Evolutionary Algorithms with Dynamic Population Size and Local Exploration for Multiobjective Optimization," *IEEE Transactions on Evolutionary Computation*, vol. 5, no. 6, pp. 565–588, December 2001.

[8] David A. Van Veldhuizen, *Multiobjective Evolutionary Algorithms: Classifications, Analyses, and New Innovations*, Ph.D. thesis, Department of Electrical and Computer Engineering. Graduate School of Engineering. Air Force Institute of Technology, Wright-Patterson AFB, Ohio, May 1999.

[9] David A. Van Veldhuizen and Gary B. Lamont, "Multiobjective Evolutionary Algorithm Research: A History and Analysis," Tech. Rep. TR-98-03, Department of Electrical and Computer Engineering, Graduate School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, Ohio, 1998.

[10] Jason R. Schott, "Fault Tolerant Design Using Single and Multicriteria Genetic Algorithm Optimization," M.S. thesis, Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, Cambridge, Massachusetts, May 1995.

[11] Kalyanmoy Deb, Lothar Thiele, Marco Laumanns, and Eckart Zitzler, "Scalable Multi-Objective Optimization Test Problems," in *Congress on Evolutionary Computation (CEC'2002)*, Piscataway, New Jersey, May 2002, vol. 1, pp. 825–830, IEEE Service Center.

[12] Kalyanmoy Deb, "Multi-Objective Genetic Algorithms: Problem Difficulties and Construction of Test Problems," *Evolutionary Computation*, vol. 7, no. 3, pp. 205–230, Fall 1999.

[13] W. Habenicht, "Quad trees: A data structure for discrete vector optimization problems," in *Lecture Notes in Economics and Mathematical Systems No. 209*, 1982, pp. 136–145.

[14] Carlos A. Coello Coello, "Handling Preferences in Evolutionary Multiobjective Optimization: A Survey," in *2000 Congress on Evolutionary Computation*, Piscataway, New Jersey, July 2000, vol. 1, pp. 30–37, IEEE Service Center.