Propuesta de un algoritmo evolutivo aplicado a problemas de optimización

Javier Izetta Riera[1] y Nilda M. Pérez Otero[1]

[1] Grupo de Investigación y Desarrollo en Informática Aplicada, Facultad de Ingeniería,
Universidad Nacional de Jujuy, Ítalo Palanca 10
4600 S. S. de Jujuy, Argentina
{javierizetta, nilperez}@gmail.com

Resumen. En las últimas décadas diferentes tipos de algoritmos de optimización han sido desarrollados para resolver una gran cantidad de problemas. El principal desafío se presenta cuando el problema presenta una función objetivo altamente no lineal y no convexa, esto dificulta garantizar la localización del mínimo global. Por lo tanto, la necesidad de encontrar nuevas metaheurísticas que proporcionen un mejor desempeño en este tipo de problemas de optimización sigue aún vigente. En este trabajo se presenta el resultado de modificar AEvol, un algoritmo genético con un operador sencillo. La modificación consiste en la incorporación de un operador de cruza: BLX-α. Además de la incorporación de tres criterios de selección de padres: ruleta, torneo y muestreo estocástico universal. Al validar el nuevo algoritmo utilizando seis funciones *benchmarks* con diferentes configuraciones de parámetros, se obtuvieron resultados satisfactorios.

Palabras Clave: Problemas de optimización**,** Computación evolutiva, Metaheurísticas.

1 Introducción

La computación evolutiva es una rama de la inteligencia artificial, inspirada en los mecanismos de evolución biológica, que involucra problemas de optimización global. Está considerada como una técnica metaheurística y por lo tanto es adecuada para la resolución de problemas de optimización con espacios de búsqueda extensos, dispersos, no lineales y no convexos. Dentro de esta rama se destacan los algoritmos evolutivos (EA), una familia de métodos relacionados con la optimización metaheurística inspirados en la teoría de la evolución que fueron introducidos por primera vez en el trabajo de John Henry Holland [1]. Estos métodos y han demostrado ser adecuados en problemas de mayor complejidad y envergadura en donde otros métodos no son capaces de encontrar soluciones en un tiempo razonable.

Por otro lado, aunque se desarrollaron y aplicaron numerosos métodos estocásticos y determinísticos de optimización global, todos presentan ciertas ventajas y desventajas. Incluso los métodos de optimización estocásticos, que han resultado más prometedores que los determinísticos, presentan limitaciones significativas para hallar

el óptimo global forzando a negociar la velocidad y la confiabilidad de los resultados obtenidos.

Entre los métodos estocásticos, los AE son muy utilizados debido a su eficiencia para proporcionar soluciones apropiadas/adecuadas con un bajo costo computacional. Sin embargo, está aún vigente la necesidad de encontrar nuevas metaheurísticas que proporcionen un mejor desempeño en problemas de optimización de Funciones Objetivo que tienden a concentrarse en mínimos locales, disminuyendo la probabilidad de hallar un mínimo global.

En un primer acercamiento a este problema, el Grupo de Investigación y Desarrollo en Informática Aplicada (GIDIA) de la Facultad de Ingeniería de la Universidad Nacional de Jujuy, como parte de su estudio sobre técnicas de inteligencia artificial aplicadas a la optimización en modelos de termodinámica, propuso un algoritmo evolutivo con operador genético sencillo llamado AEvol [2], aplicado al cómputo de equilibrio de fases en sistemas termodinámicos de gran relevancia en procesos industriales en general, y en procesos de separación en particular [3], obteniendo buenos resultados.

La primera implementación de este algoritmo tuvo un comportamiento adecuado en los cálculos de mínimos de las funciones benchmarks seleccionadas y proporcionó resultados satisfactorios en los casos de estudio propuestos hasta el momento. Sin embargo, el interés en lograr una mejor relación entre la exploración y explotación del área de búsqueda motivó la realización de una nueva implementación de AEvol [2].

En el presente trabajo se introduce una variante de la implementación de AEvol que incorpora el operador de cruza BLX-α para la combinación de la información genética [4], con el objetivo de ampliar la búsqueda del espacio de soluciones y mejorar la tasa de éxitos disminuyendo la probabilidad de caer en mínimos locales y lograr independencia de los valores de inicialización del cómputo.

Para la evaluación y validación del nuevo algoritmo, se utilizaron seis funciones benchmarks propuestos para el Congreso de Computación Evolutiva de 2008 [CEC'08] que se encuentran en el trabajo de Tang y colegas [5]. Los resultados obtenidos resultaron satisfactorios.

El resto de este artículo se estructura como sigue: en la parte 2 se presentan los antecedentes del algoritmo y su nueva estructura, en la parte 3 se describen las funciones en los que se aplica el algoritmo propuesto, en la parte 4 se presenta el diseño de los experimentos y un análisis de resultados y en la parte 5 se muestran las conclusiones.

2 Algoritmo propuesto

2.1 Algoritmos evolutivos

Los AE son algoritmos que pertenecen a la rama de las metaheurísticas poblacionales que han sido aplicados exitosamente en la optimización de una gran variedad de problemas de optimización en dominios como planificación, diseño, identificación y simulación, control y clasificación, entre otros [6].

Estos algoritmos utilizan el proceso de aprendizaje colectivo de una población de individuos. Usualmente, cada individuo representa una posible solución dentro del espacio de soluciones de un problema dado. Los descendientes de los individuos se generan por medio de procesos aleatorios que modelan la mutación y la cruza. La mutación corresponde a una alteración o cambio de las características de un mismo individuo, mientras que la cruza intercambia información entre dos o más individuos existentes. Para asignar una medida de adaptación a los individuos se evalúa su comportamiento en el entorno. Tomando como base esta medida de adaptación el proceso de selección favorece la reproducción de los mejores individuos. [7]. En la Fig. 1 se presenta la estructura general de algoritmo evolutivo según se presenta en [5], en la etapa de reproducción se producen nuevas generaciones mediante diferentes operadores (cruza, mutación, entre otros).

```
Generar(P(0)) ; /*Población Inicial*/
t= 0;
Mientras no CriterioDeCorte(P(t)) hacer
Evaluar(P(t));
    P'(t)   = Selección(P(t));
    P'(t)   = Reproducción(P'(t)); Evaluar(P'(t));
    P(t+1) = Reemplazar(P(t), P'(t));
    t= t + 1;
Fin Mientras
Salida Mejor individuo o mejor población encontrada.
```

Fig. 1. Estructura general de un algoritmo evolutivo.

2.2 AEvol

Siguiendo la línea de la Estrategia Evolutiva [8], donde no se usan operadores de cruza, en AEvol sólo se aplica mutación para la generación de nuevas soluciones. Esto se hace mediante la generación de nuevas soluciones a partir de la definición de hipercuadrados en el espacio de búsqueda. En cada iteración i se generan n nuevas soluciones a partir de la mejor solución obtenida en la iteración $i-1$. Para construir cada una de las soluciones se utilizan m hipercuadrados concéntricos, con centro en las coordenadas de la mejor solución usando el método de partición geométrica, de acuerdo a la ecuación (1). La estructura de los hipercuadrados se muestra en la fig. 2.

$$sl_j = \frac{sl_m}{2^{m-j}} para\ j = 1,2,\dots,m \quad \textbf{(1)}$$

Donde sl_j es el semilado de cada uno de los m hipercuadrados y m, la cantidad de hipercuadrados. Una vez construidos los m hipercuadrados se genera aleatoriamente un punto dentro de cada uno de ellos (zona 1, zona 2 y zona 3 en la Fig. 2). Se evalúa la función objetivo para cada uno de esos puntos y, finalmente, se agrega la mejor de las m soluciones a la lista de las k soluciones de la iteración $i-1$. El valor inicial del semilado sl_m, es un parámetro del algoritmo que se modifica en tiempo de ejecución en función de una variable que indica cuántas iteraciones han transcurrido sin obtener una mejor solución.

La validación de AEvol se presentó en [2], en el mismo artículo se muestra la aplicación del algoritmo en la minimización de la función de Gibbs, función

termodinámica relacionada al equilibrio de fases, en tres diferentes sistemas. Si bien el algoritmo se comportó de manera adecuada en los cálculos de mínimos de las funciones multimodales seleccionadas como benchmarks y los resultados obtenidos en los casos de estudio presentados se consideraron satisfactorios, se propuso la inclusión un operador de cruza, con el propósito de mejorar la tasa de éxitos. En el siguiente apartado se muestra cómo se modificó a AEvol al incorporar un operador de cruza.

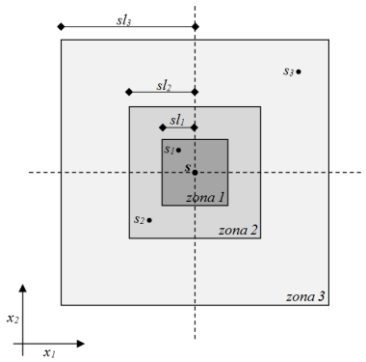

Fig. 2. Tres hipercuadrados en 2 dimensiones, cada uno conteniendo un punto generado aleatoriamente tomando como base el punto s.

2.3 Incorporación del operador de cruza a AEvol

En el diseño de una metaheurística se aconseja tener en cuenta dos criterios contradictorios, la diversificación (exploración del espacio de búsqueda) y la intensificación (explotación de las mejores soluciones halladas) [6].

Al utilizar sólo un operador de mutación, en AEvol se priorizaba la diversificación sobre la intensificación. Con el objetivo de agregar intensificación al algoritmo se incorporó un operador de cruza. El operador seleccionado fue el *BLX-alpha* propuesto por [4]. A su vez se incorporaron tres criterios de selección de padres: ruleta, torneo y muestreo estocástico universal (*stochastic universal sampling* - SUS).

La Fig. 3 muestra la estructura de AEvol con el operador de cruza.

```
Generar k soluciones aleatorias; /*Población Inicial*/
Para (i=0; i<iterTot; i++)
   Calcular FO para las k soluciones;
   Rankear las k soluciones por valor de FO;
   nc = nc de la mejor solución;
   Si nc > noCambia
        Cruza(k soluciones);
   Sino
        Mutación(mejor solución);
   Fin Si
Fin Para
Salida Mejor individuo o mejor población encontrada.
```

Fig. 3. Estructura de AEvol con el operador de cruza.

Parámetros. A los parámetros de AEvol original: *Sol*, cantidad de soluciones iniciales; *iterTot*, cantidad máxima de iteraciones, *cantHiper*, cantidad de hipercudrados y *sl*, tamaño del semilado inicial; se agregan las variables *cantHijos* y *noCambia*, que indica el número máximo de veces que se repetirá la mejor solución obtenida en el proceso de mutación antes de realizar el proceso de cruza. Cada una de las *k* soluciones tiene una variable llamada *no_cambia* que se inicializa en 1 y se incrementa cada vez que esa solución es rankeada como la mejor. En el caso que *noCambia* es igual a 0 (cero), el algoritmo sólo utiliza el operador de cruza.

3 Funciones de prueba

Para el proceso de evaluación del algoritmo AEvol con el operador de cruza, se tomaron las primeras 6 funciones de prueba provenientes del CEC'2008 [5]. Las dos primeras funciones son unimodales y las restantes cuatro son multimodales. En la Tabla 1 se presentan las funciones usadas para evaluar la performance del algoritmo propuesto.

Tabla 1. Descripción de las funciones de prueba utilizadas

Función	Nombre	N° de Variables	Dominio de búsqueda	Mínimos Locales	Mínimo global
1	Shifted Sphere	2	$-100<x_i<100$	-	-450
2	Shifted Schwefel's Problem 2.21	2	$-100<x_i<100$	-	-450
3	Shifted Rosenbrock	2	$-100<x_i<100$	Varios	390
4	Shifted Rastrigin	2	$-5<x_i<5$	Varios	-330
5	Shifted Griewank	2	$-600<x_i<600$	Varios	-180
6	Shifted Ackley	2	$-32<x_i<32$	Varios	-140

Para cada una de las funciones, se empleó el algoritmo evolutivo modificado con un operador de cruza con el propósito de encontrar el mínimo global correspondiente en $\boldsymbol{o}=\{o_1, o_2\}$.

4 Experimentación y Resultados

4.1 Experimentación

La aplicación de AEvol modificado a las funciones de prueba tiene como objetivo estudiar el desempeño del algoritmo tras la incorporación del operador de cruza y evaluar su rendimiento respecto a diferentes configuraciones de los parámetros en los tres métodos de selección implementados. Se tomó como criterio de evaluación el error medio de 500 ejecuciones del método. El cálculo del error está determinado por la ecuación 2.

$$\text{Error} = F(x) - F(x') \quad \textbf{(2)}$$

Donde $x' = o$, es el óptimo global.

Para el análisis se formularon veintiún experimentos, en la tabla 2 se muestra la configuración de cada uno de ellos.

En primer lugar para determinar el número de iteraciones se evaluó el desempeño del método AEvol modificado sobre los casos de estudio seleccionados para 500, 1000 y 10000 iteraciones. Los resultados determinaron que no se obtiene una mejora significativa al aumentar el número de iteraciones a partir de las 500, sin embargo se pudo observar una leve mejora en algunos casos con 1000 iteraciones, es por eso que se decidió realizar el análisis de los demás parámetros con 1000 iteraciones.

Tabla 2. Experimentos realizados para cada caso de estudio

Experimento	NoCambia	Selección (sel)	Sol	sl
E(Id1,NoCambia,sel)	{5,10,20,50}	{Ruleta, Torneo, SUS}	10	20
E(Id2, NoCambia,sel)	{5,10,20,50}	{Ruleta, Torneo, SUS}	10	2
E(Id3, NoCambia,sel)	{5,10,20,50}	{Ruleta, Torneo, SUS}	10	1
E(Id4, NoCambia,sel)	{5,10,20,50}	{Ruleta, Torneo, SUS}	20	20
E(Id5, NoCambia,sel)	{5,10,20,50}	{Ruleta, Torneo, SUS}	20	2
E(Id6, NoCambia,sel)	{5,10,20,50}	{Ruleta, Torneo, SUS}	20	1
E(Id7, NoCambia,sel)	{5,10,20,50}	{Ruleta, Torneo, SUS}	30	20
E(Id8, NoCambia,sel)	{5,10,20,50}	{Ruleta, Torneo, SUS}	30	2
E(Id9, NoCambia,sel)	{5,10,20,50}	{Ruleta, Torneo, SUS}	30	1
E(Id10,sol,sel)	5	{Ruleta, Torneo, SUS}	{10,20,30}	20
E(Id11,sol,sel)	5	{Ruleta, Torneo, SUS}	{10,20,30}	2
E(Id12,sol,sel)	5	{Ruleta, Torneo, SUS}	{10,20,30}	1
E(Id13,sol,sel)	10	{Ruleta, Torneo, SUS}	{10,20,30}	20
E(Id14,sol,sel)	10	{Ruleta, Torneo, SUS}	{10,20,30}	2
E(Id15,sol,sel)	10	{Ruleta, Torneo, SUS}	{10,20,30}	1
E(Id16,sol,sel)	20	{Ruleta, Torneo, SUS}	{10,20,30}	20
E(Id17,sol,sel)	20	{Ruleta, Torneo, SUS}	{10,20,30}	2
E(Id18,sol,sel)	20	{Ruleta, Torneo, SUS}	{10,20,30}	1
E(Id19,sol,sel)	50	{Ruleta, Torneo, SUS}	{10,20,30}	20
E(Id20,sol,sel)	50	{Ruleta, Torneo, SUS}	{10,20,30}	2
E(Id21,sol,sel)	50	{Ruleta, Torneo, SUS}	{10,20,30}	1

Para cada caso de estudio y cada experimento se calcularon los valores estadísticos media, desviación estándar, 1er cuartil, mediana, 3er cuartil, peor y mejor obtenidos al considerar los resultados de 500 ejecuciones independientes del método. Apuntando el análisis al parámetro no cambia se tomó en cuenta los resultados de los experimentos E(Id1,NoCambia,sel) al E(Id9,NoCambia,sel). A modo de ejemplo en la Tabla 3 y en la Fig. 4 se presenta el resumen de resultados de los experimentos E(Id1,NoCambia,sel) al E(Id9,NoCambia,sel) realizados para función 1. Los valores paramétricos del algoritmo considerados fueron: *canHiper*=3, *cantHijos*=5. De igual manera para el análisis del parámetro *Sol* se tomó en cuenta los resultados de los experimentos E(Id10,Sol,sel) al E(Id21,Sol,sel). A modo de ejemplo en la Tabla 4 y la Fig. 5 se presenta el resumen de resultados de los experimentos E(Id10,Sol,sel) al E(Id21,Sol,sel) realizados para el función 1.

Tabla 3. Resultados obtenidos para los experimentos E(Id1,NoCambia,sel) al E(Id9,NoCambia,sel) de la función 1

	Error Medio nc=5	Error Medio nc=10	Error Medio nc=20	Error Medio nc=50
Media	4,831453	0,858607	0,227912	0,458339
SD	8,715856	1,064892	0,160064	0,870460
1er Cuartil	0,459885	0,060544	0,056928	0,084758
Mediana	0,543995	0,234636	0,205013	0,256951
3er Cuartil	3,430151	1,861378	0,381993	0,416539
Peor	28,163318	2,896187	0,504216	0,549457
Mejor	0,069440	0,044863	0,041006	0,041178

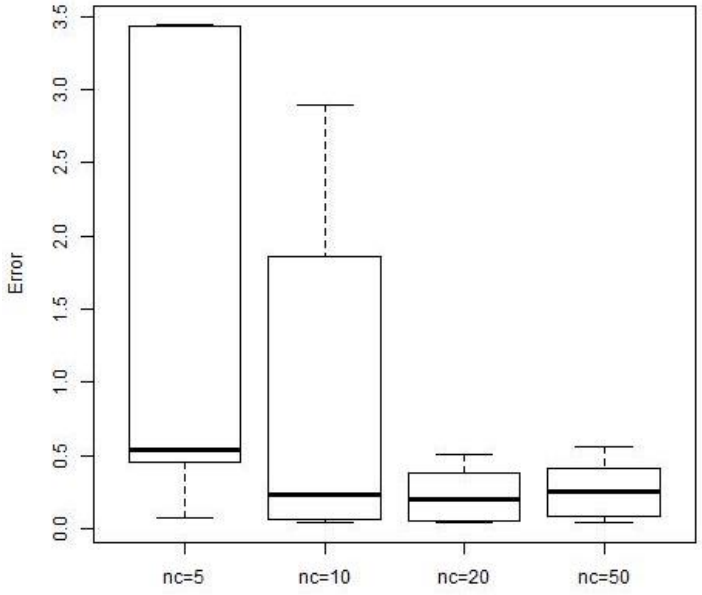

Fig. 4. Diagrama de cajas de los experimentos E(Id1,NoCambia,sel) al E(Id9,NoCambia,sel). Los outliers no son representados en el diagrama para una mejor visualización.

Tabla 4. Resultados obtenidos para los experimentos E(Id10,Sol,sel) al E(Id21,Sol,sel) de la función 1

	Error Medio sol=10	Error Medio sol=20	Error Medio sol=30
Media	0,656272	2,819905	1,093330
SD	0,969904	7,747661	2,265103
1er Cuartil	0,085443	0,067279	0,081235
Mediana	0,255417	0,219824	0,220730
3er Cuartil	0,543995	0,902320	0,459885
Peor	3,440846	28,163318	8,067588
Mejor	0,041006	0,044737	0,041178

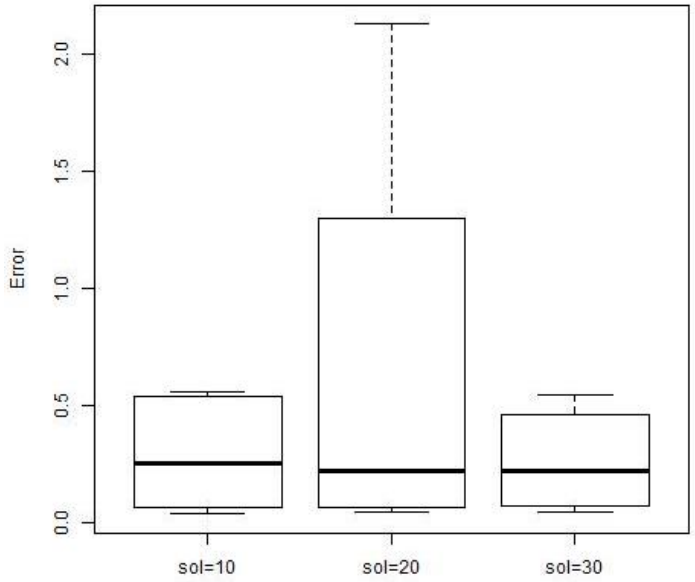

Fig. 5. Diagrama de cajas de los experimentos E(Id10,sol,sel) al E(Id2,sol,sel). Los outliers no son representados en el diagrama para una mejor visualización.

4.2 Análisis de resultados

Los experimentos permitieron determinar la mejor configuración de parámetros para cada una de las funciones, en la Tabla 5 se presentan estas configuraciones.

Tabla 5. Mejores configuraciones del algoritmo para cada función.

Función	NoCambia	Sol	sl
1	20	10	1
2	10	10	1
3	5	30	20
4	5	10	1
5	5	10	1
6	5	10	1

Para el análisis de exactitud del método se consideraron los mejores errores alcanzados para las configuraciones presentadas en la Tabla 3. La Tabla 6 presenta los resultados obtenidos para cada CE.

5 Conclusiones

En este trabajo se presentó la modificación de AEvol, un algoritmo evolutivo con un

operador de mutación sencillo. La modificación consistió en agregar el operador de cruza *BLX-alpha*.

Se aplicó AEvol modificado a las funciones tomadas del CEC´08 para estudiar el desempeño del algoritmo tras la incorporación del operador de cruza y evaluar su rendimiento respeto a diferentes configuraciones de los parámetros en los tres métodos de selección implementados.

Los resultados obtenidos demuestran que se logró un buen desempeño del algoritmo al implementar el operador de cruza, además se pudo determinar la mejor configuración de parámetros para cada función. Sin embargo, en vista de que se necesitó una configuración diferente de los parámetros del algoritmo para cada función, se considera interesante seguir investigando sobre estas características.

Tabla. 6. Los mejores errores alcanzados para las configuraciones presentadas en la Tabla 3 del algoritmo propuesto para cada función

CE	Ruleta	Torneo	SU
1	**0,041006**	0,041785	0,041362
2	**0,024305**	0,029768	0,026412
3	**8,180332**	8,842712	10,365095
4	**0,000175**	0,000199	0,000198
5	0,008598	**0,007792**	0,009127
6	0,016695	0,016205	**0,015883**

Referencias

1. Holland, J. H.. Adaptation in Natural ans Artificial Systems. University of Michigan Press, Ann Arbor. (1975). Republicado por MIT Press en 1992.
2. Pérez Otero, N., Zacur J. L.. Algoritmo evolutivo con un operador genético sencillo aplicado al cómputo de equilibrio de fases en sistemas termodinámicos. XVIII Congreso Argentino de Ciencias de la Computación. (CACIC 2012). ISBN 978-987-1648-34-4. Universidad Nacional del Sur. Bahía Blanca. (2012).
3. Zhang, H., Bonilla-Petriciolet, A., G. P. Rangaiah, A. A Review on Global Optimization Methods for Phase Equilibrium Modeling and Calculations. The Open Thermodynamics Journal. Vol. 5, (Suppl 1-M7) pp. 71–92. (2011)
4. Eshelman L. J., Schaffer J. D. Real-Coded Genetic Algorithms and Interval-Schemata. Foundation of Genetic Algorithms 2, L.Darrell Whitley (Ed.) (Morgan Kaufmann Publishers, San Mateo), 187–202. (1993).
5. Tang, K., Yáo, X., Suganthan, P. N., MacNish, C., Chen, Y. P., Chen, C. M., & Yang, Z. Benchmark functions for the CEC'2008 special session and competition on large scale global optimization. Nature Inspired Computation and Applications Laboratory, USTC, China. (2007).
6. Talbi, E. G. Metaheuristics: *From design to implementation*. ISBN 978-0-470-27858-1. Wiley Eds. (2009)
7. Back, T., Fogel, D. B., Z. Michalewicz, (Eds.) Handbook of Evolutionary Computation (1st ed.). IOP Publ. Ltd., Bristol, UK, UK. Capítulo: B1.1 de Thomas Back. (1997).
8. Rechenberg, I. Cybernetic solution path of an experimental problem. Technical Report, Royal Aircraft Establishment Library Translation No. 1112, Farnborough, UK. (1965).