Algoritmos Genéticos: Una aproximación a la Solución de Problemas Large Scale Global Optimization Propuestos en CEC 2013

Edwin Sneyder Gantiva Ramos[1]

Universidad Nacional de Colombia, Bogotá, Colombia,
I.esgantivar@unal.edu.co

Abstract. El presente documento tiene como fin principal exponer técnicas basadas en estrategias evolutivas [2] con el objetivo de estimar soluciones óptimas para las funciones propuestas en CEC 2013 [4]. En este documento se hará una descripción de los operadores genéticos propuestos, estrategias de selección y estrategias de reemplazo, así como todos los parámetros que se ven relacionados durante la evolución de los individuos generación tras generación. La propuesta principalmente se basa en operadores genéticos auto adaptativos [3], que es una técnica que ha presentado buenos resultados con respecto a la aplicación de algoritmos genéticos convencionales.

Keywords: Optimización, Algoritmos Genéticos, Operadores Genéticos Auto Adaptativos, Large Scale Global Optimization, Estrategias Evolutivas

1 Introducción

Los Algoritmos Genéticos son técnicas de optimización inspirados en los principios de la evolución natural Darwiniana, dichos algoritmos constituyen un campo de estudio de la computación evolutiva y presentan un tono particular y llamativo, puesto que en medio de toda la complejidad que está en torno a ellos la idea subyacente puede dar una percepción de ser intuitiva.

Los Algoritmos Genéticos han presentado una alternativa a la solución de problemas de optimización que poseen altas dimensiones, dicha solución consiste en hacer una aproximación a los óptimos de los diferentes problemas que pueden ser abordados.

El presente documento tendrá la siguiente estructura, estará compuesto por cinco secciones, en la primera se hará una presentación de los resultados de referencia que se encuentran presente dentro de la documentación de la convocatoria de CEC 2013 , en la siguiente sección se desarrollara una somera revisión acerca de los conceptos principales que se hace necesario conocer alrededor de los algoritmos genéticos, seguido de esta se presentaran las propuestas para los

problemas que se han seleccionado, casi finalizando se presentaran los principales resultados encontrados al realizar las diferentes experimentaciones y para terminar se tendrá una sección que contendrá las conclusiones que ha dejado el desarrollo de este trabajo.

2 Estado del Arte

El principal desarrollo que tenemos alrededor de los problemas de optimización propuestos en [4] son los mismos que se presentan como punto de partida dentro de la convocatoria para la solución de dichos problemas. En las siguientes tablas se muestran los resultados obtenidos con el algoritmo DECC-G [5].

Algorithms	**Quality**	f1	f2	f3	f4	f5	f6
DECC-G	Best	1.75e-13	9.90e+02	2.63e-10	7.58e+09	7.28e+14	6.96e-08
	Median	2.00e-13	1.03e+03	2.85e-10	2.12e+10	7.28e+14	6.08e+04
	Worst	2.45e-13	1.07e+03	3.16e-10	6.99e+10	7.28e+14	1.10e+05
	Mean	2.03e-13	1.03e+03	2.87e-10	2.60e+10	7.28e+14	4.85e+04
	Std	1.78e-14	2.26e+01	1.38e-11	1.47e+10	1.51e+05	3.98e+04
Algorithms	**Quality**	f7	f8	f9	f10	f11	f12
DECC-G	Best	1.96e+08	1.43e+14	2.20e+08	9.29e+04	4.68e+10	9.80e+02
	Median	4.27e+08	3.88e+14	4.17e+08	1.19e+07	1.60e+11	1.03e+03
	Worst	1.78e+09	7.75e+14	6.55e+08	1.73e+07	7.16e+11	1.20e+03
	Mean	6.07e+08	4.26e+14	4.27e+08	1.10e+07	2.46e+11	1.04e+03
	Std	4.09e+08	1.53e+14	9.89e+07	4.00e+06	2.03e+11	5.76e+01
Algorithms	**Quality**	f13	f14	f15			
DECC-G	Best	2.09e+10	1.91e+11	4.63e+07			
	Median	3.36e+10	6.27e+11	6.01e+07			
	Worst	4.64e+10	1.04e+12	7.15e+07			
	Mean	3.42e+10	6.08e+11	6.05e+07			
	Std	6.41e+09	2.06e+11	6.45e+06			

3 Marco Teórico

En esta sección abordaremos conceptos básicos de algoritmos genéticos que son necesarios para el correcto entendimiento de la propuesta que se hará alrededor de la solución de los problemas presentados en [4].

Algoritmo Genético Convencional:

Algorithm 3.1: Algoritmo Genetico($f, \mu, CondicionDeTerminacion$)

1 Inicializar la Población Inicial P_0;
2 Evaluar(P_0, f);
3 **while** $CondiciondeTerminacion$ **do**
4 $\quad descendientes = GENERACION(P_t, f, SELECCION)$;
5 $\quad P_t = FUNCION\_REEMPLAZO(descendientes, P_{t-1})$

.

Función de Fitness: Esta función representara en un valor numérico el rendimiento del individuo dentro del problema, esta definida de acuerdo a los 15 problemas que se estudiaran en este documento.

Operador Genético Auto Adaptativo: Un operador genético es una operación que se aplica generación a generación sobre un individuo, dentro de un algoritmo genético convencional generalmente se aplica un proceso de cruce (Combinación entre dos o más Parientes) seguido de una mutación que se dará de acuerdo a un parámetro de probabilidad. Sin embargo, existe un tipo de algoritmo genético en donde sobre cada individuo en una generación se aplica únicamente un operador, la elección de este operador está asociado a una rata que se encuentra codificada dentro del cromosoma del individuo, pero lo anterior hace que se tengan que dar muchos más parámetros dentro de la inicialización del algoritmo genético, por esto J. Gomez expone una técnica en [3] que ha sido la inspiración para el desarrollo de este problema, presentando mejores resultados respecto a la aplicación de algoritmos genéticos convencionales.

Estrategia de Selección: En los algoritmos genéticos la tarea de escoger a los individuos constituye una tarea de especial importancia, por esto han sido desarrolladas, así:

1. Elitista
2. Aleatoria
3. Ruleta
4. Torneo

La primera estrategia esta basada en solo escoger y dar oportunidad a los individuos que posean el mejor fitness. La segunda estrategia elige a la población de individuos candidatos a ser padres de una forma aleatoria siguiendo una distribución de probabilidad uniforme. La tercera y cuarta aunque se basan en el fitness del individuo para definir la población de candidatos a ser padres, permite que individuos que no tengan un fitness tan bueno tengan la posibilidad de entrar a esta población, haciendo la aclaración que un individuo que tenga mejor fitness siempre tendrá mas posibilidad de ser escogido para hacer parte de la población de candidatos.

Estrategia de Reemplazo: En esta tarea se tienen dos principales estrategias, así:

1. Estado Estable
2. Elitista

En la primera la caracteristica principal, es que el descendiente de un individuo podra tomar su lugar dentro de la población si y solo si su fitness es mejor que el de su padre. En la segunda estrategia la población de descendientes y de padres son unidas y solo son escogidos para pasar a la siguiente generación los primeros n individuos donde n es el tamaño dado de la población.

4 Propuesta

4.1 Generación de la Población Inicial

La población inicial ha tenido un tamaño por defecto de 100 individuos, estos de acuerdo a los parámetros fijados en [4] tienen una dimensión de 1000, y los genes del cromosoma están representados en valores reales de acuerdo a los rangos planteados. Esta población fue generada de forma aleatoria siguiendo una distribución de probabilidad uniforme.

4.2 Estrategia de Selección

De acuerdo a la información presentada en la anterior sección se ha propuesto para la solución de los problemas una estrategia de selección de tipo ruleta, que se aplicara al principio de cada nueva generación para generar un conjunto de cadidatos elegibles, para que dado el caso que la ariedad del operador seleccionado sea mayor que 1, de esta lista de candidatos sean elegidos cuanto sean necesarios de una manera aleatoria.

4.3 Estrategia de Reemplazo

Se asume una estrategia de reemplazo de estado estable convencional.

4.4 Operadores

Se han propuesto los siguientes operadores que son aplicables cuando la naturaleza de los genes del cromosoma es de tipo real.

- **Operadores de Mutación**: Para estos operadores se recibe un parámetro de mutación que esta asociado a un porcentaje de los genes que serán objeto de mutación, estos genes son escogidos de acuerdo a la dimensión del cromosoma y siguiendo una distribución de probabilidad uniforme.

  1. Mutación Gaussiana: Este operador esta basado en la generación de números gaussianos alrededor de un parámetro σ, y usando las estrategias evolutivas[2] se generan para cada gen que va a ser objeto de mutación.
  2. Mutación Uniforme: Este operador usa una distribución uniforme para la generación de un factor de mutación, ademas de esto de manera aleatoria escoge si el gen mutara hacia el limite inferior o superior del espacio factible

- **Operadores de Cruce**: Este tipo de operadores necesitan de al menos dos parientes para generar su descendencia, ademas de esto es aplicado a la totalidad de la dimensión de los cromosomas de los individuos

- AllXOver: De manera aleatoria siguiendo una distribución de probabilidad uniforme distribuye para dos descendientes los genes de forma excluyente, en otras palabras si el valor del gen es de un descendiente no puede pertenecer a otro descendiente.
- PivotXOver: De manera aleatoria siguiendo una distribución de probabilidad uniforme se escoge un numero dentro del rango de la dimensión del cromosoma y a partir de este *pivot* se hace una transposición de los genes de los parientes en sus respectivos descendientes.
- LinearXOver: De manera aleatoria siguiendo una distribución de probabilidad uniforme se elige un factor para realizar una combinación lineal entre los genes que pertenecen a la misma posición en los cromosomas de los parientes, así este factor s es aplicado de la siguiente manera $g_n^1 = gp_n^1 * s + gp_n^2 * (1 - s)$ y $g_n^2 = gp_n^2 * s + gp_n^1 * (1 - s)$
  .

– **Operadores de Intensidad**: Ademas de lo anterior la Mutación Gaussiana ha sido parametrizada con la estrategia de intensidad de la regla de un quinto presentada en [1], que presento una ganancia en el proceso evolutivo.

4.5 Algoritmo

El algoritmo propuesto en base a los parámetros expuestos anteriormente esta basado en la propuesta de algoritmo híbrido presentada en [3], donde cada individuo tiene codificadas las ratas de los operadores en su cromosoma y estas son recompensadas o castigadas de acuerdo a su productividad.

Todos las propuestas mencionadas anteriormente se encuentran disponible para su uso y consulta en Repositorio en GitHub

5 Resultados

5.1 Experimentos para 1000 Generaciones

A continuación se presenta una tabla con el resumen de los resultados obtenidos para 1000 generaciones, así mismo se presentan las gráficas (**Fig 1 - Fig 15**) del comportamiento del fitness y los operadores a través del proceso de evolución.

Quality	f1	f2	f3	f4	f5
Mejor	3,52E+08	15966,4644	15,653379	8,27E+10	1,72E+07
Mediana	3,53E+08	15966,4644	15,653379	8,27E+10	1,72E+07
Peor	3,70E+08	22917,4648	15,952936	3,25E+11	7,26E+07
Media	3,54E+08	16557,1250	15,686614	8,71E+10	1,99E+07
Desv. Est.	2521491,963	1255,5292	0,073693	3,14E+10	9661311,965
Quality	f6	f7	f8	f9	f10
Mejor	718000,7829	1,88E+07	5,40E+15	9,74E+08	5,96E+07
Mediana	718000,7923	1,89E+07	5,40E+15	9,74E+08	5,96E+07
Peor	795076,3504	3,90E+07	2,31E+16	3,62E+09	7,96E+07
Media	720311,5491	1,91E+07	5,85E+15	1,09E+09	6,03E+07
Desv. Est.	10882,78566	2192437,07	2,60E+15	3,96E+08	3195175,986
Quality	f11	f12	f13	f14	f15
Mejor	1,64E+09	5,15E+11	1,66E+08	1,58E+09	179236,4414
Mediana	1,64E+09	5,17E+11	1,66E+08	1,58E+09	180112,7188
Peor	1,37E+10	6,18E+11	4,31E+08	3,33E+09	2019472,787
Media	1,84E+09	5,21E+11	1,72E+08	1,61E+09	214349,3034
Desv. Est.	1,26E+09	1,68E+10	3,25E+07	1,88E+08	208287,343

Tabla resumen de los resultados obtenidos al aplicar 1000 generaciones a los 15 problemas propuestos en [4]

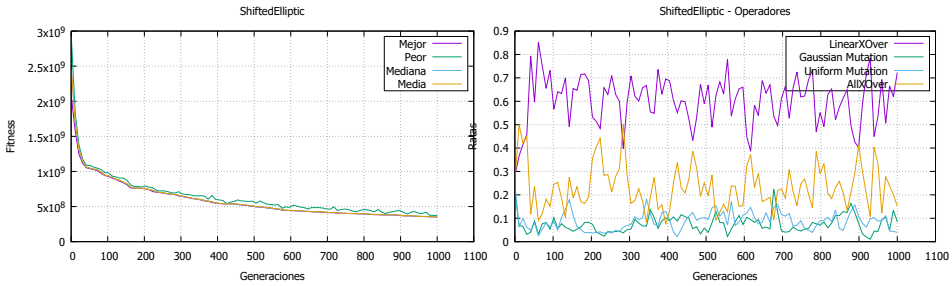

Fig. 1. Resultados para 1000 Iteraciones de la funcion f1

5.2 Experimentos para 10000 Generaciones

A continuación se presenta una tabla con el resumen de los resultados obtenidos para 10000 generaciones, así mismo se presentan las gráficas (**Fig 16 - Fig 30**) del comportamiento del fitness y los operadores a través del proceso de evolución.

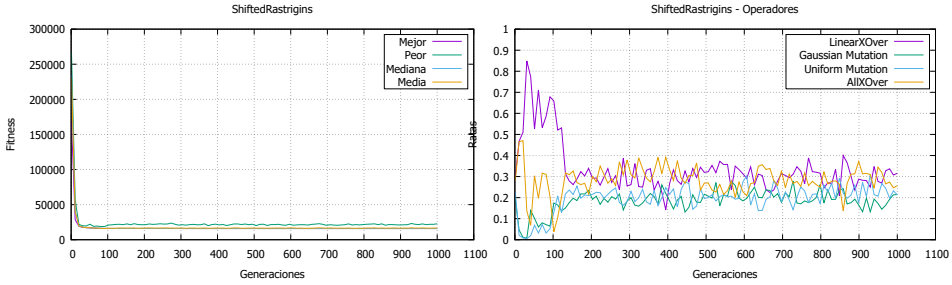

Fig. 2. Resultados para 1000 Iteraciones de la funcion f2

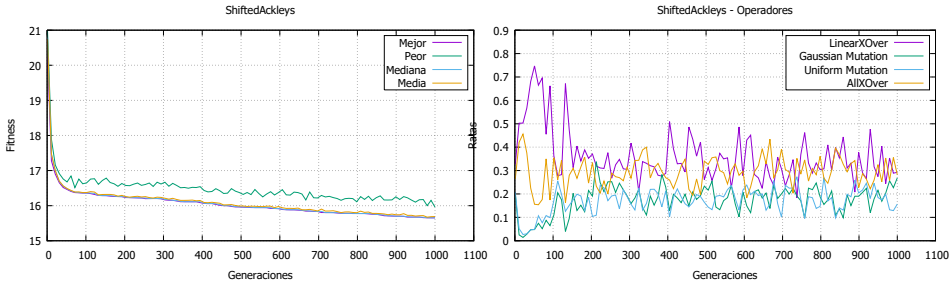

Fig. 3. Resultados para 1000 Iteraciones de la funcion f3

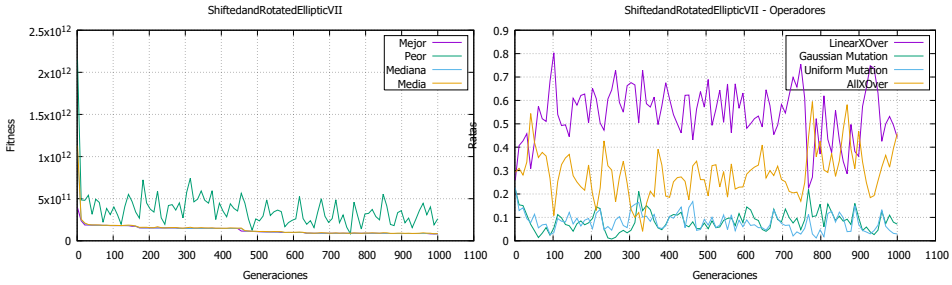

Fig. 4. Resultados para 1000 Iteraciones de la funcion f4

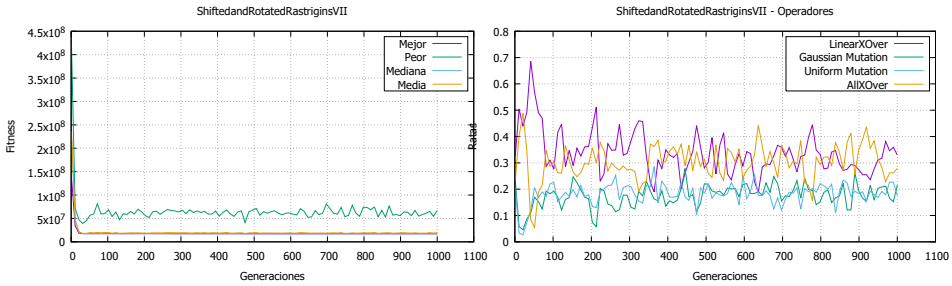

Fig. 5. Resultados para 1000 Iteraciones de la funcion f5

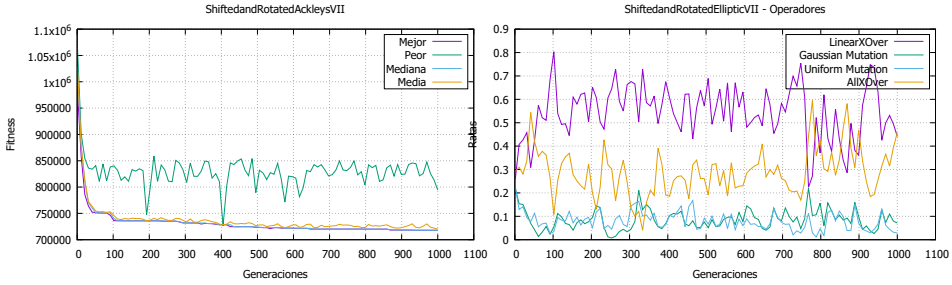

Fig. 6. Resultados para 1000 Iteraciones de la funcion f6

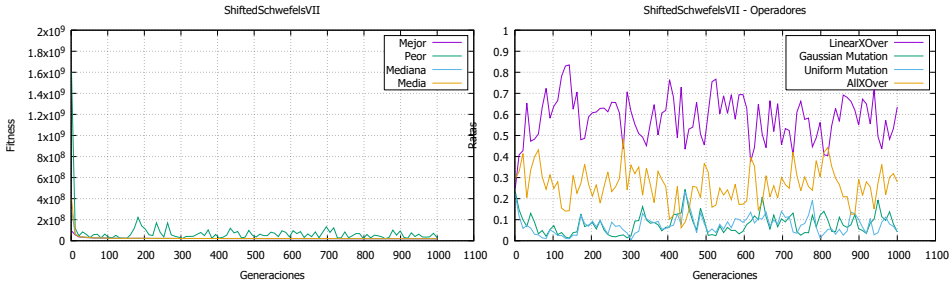

Fig. 7. Resultados para 1000 Iteraciones de la funcion f7

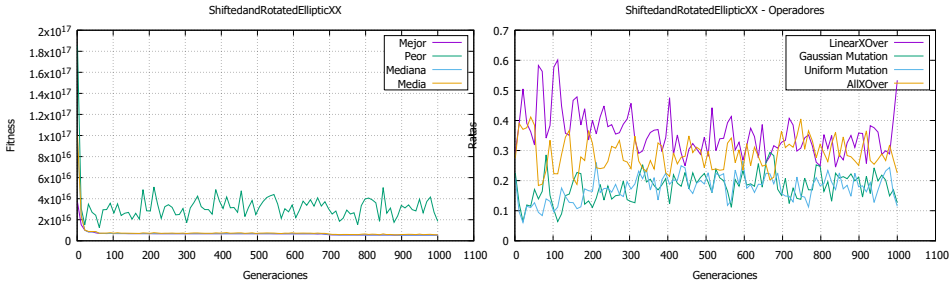

Fig. 8. Resultados para 1000 Iteraciones de la funcion f8

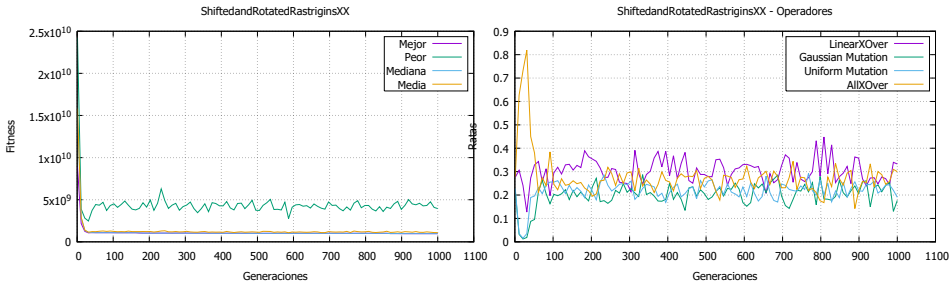

Fig. 9. Resultados para 1000 Iteraciones de la funcion f9

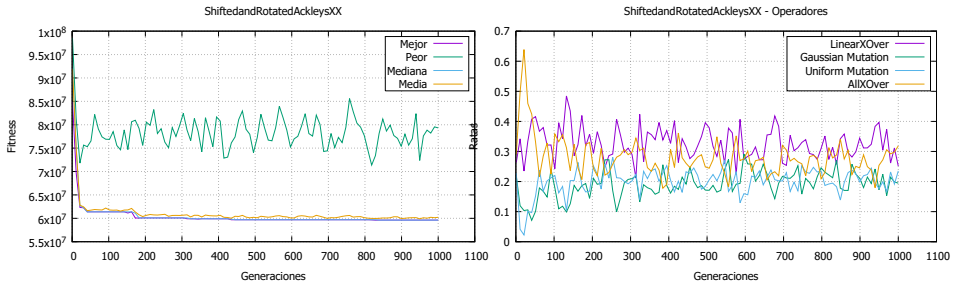

Fig. 10. Resultados para 1000 Iteraciones de la funcion f10

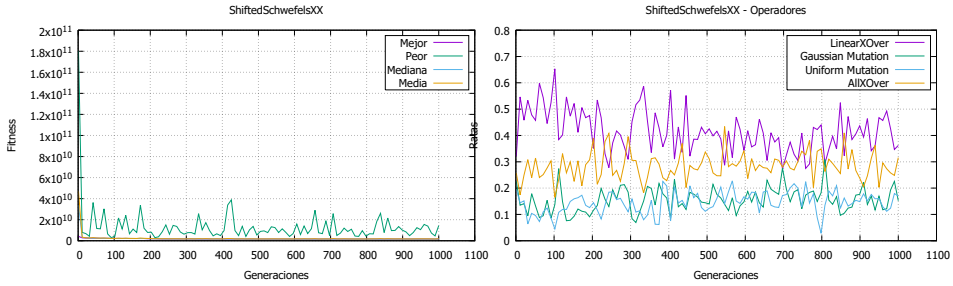

Fig. 11. Resultados para 1000 Iteraciones de la funcion f11

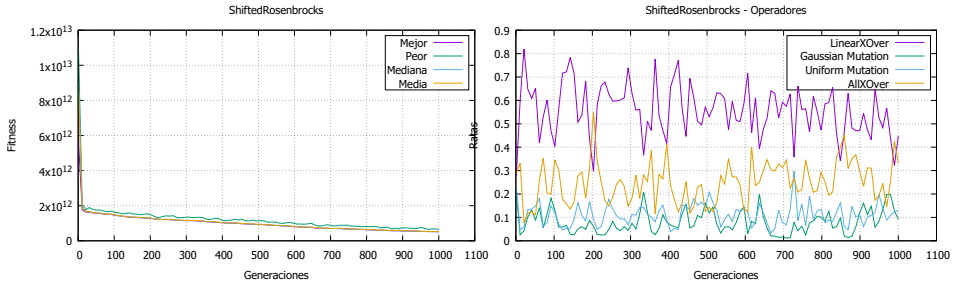

Fig. 12. Resultados para 1000 Iteraciones de la funcion f12

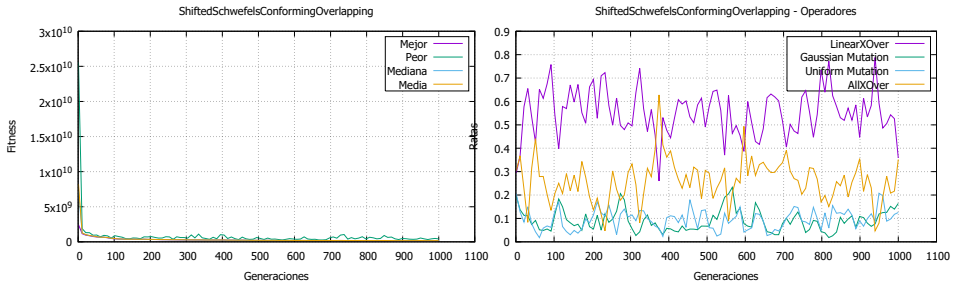

Fig. 13. Resultados para 1000 Iteraciones de la funcion f13

X

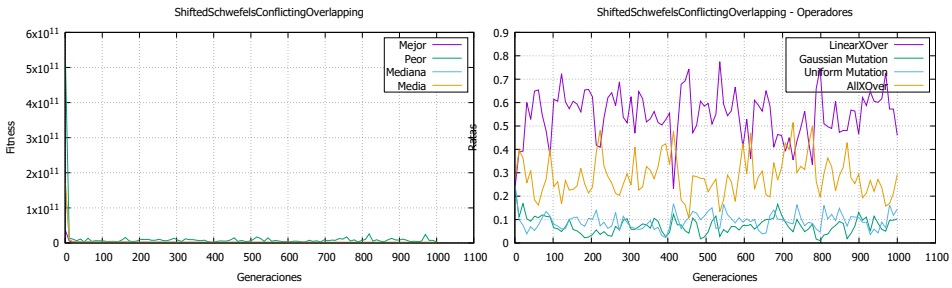

Fig. 14. Resultados para 1000 Iteraciones de la funcion f14

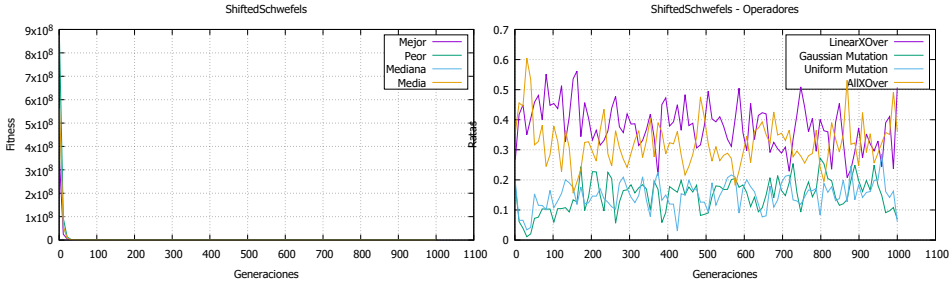

Fig. 15. Resultados para 1000 Iteraciones de la funcion f15

Quality	f1	f2	f3	f4	f5
Mejor	7.67E+07	16298.95653	13.74533399	1.14E+11	1.51E+07
Mediana	7.67E+07	16298.95653	13.75084957	1.14E+11	1.51E+07
Peor	1.26E+08	20154.08578	14.52557441	3.83E+11	6.27E+07
Media	7.86E+07	16812.41595	13.80599331	1.17E+11	1.71E+07
Desv. Est.	6903910.158	927.1059753	0.140569927	2.69E+10	7486604.217
Quality	f6	f7	f8	f9	f10
Mejor	761929.0764	1.66E+07	5.58E+15	1.06E+09	5.91E+07
Mediana	761929.0764	1.66E+07	5.58E+15	1.06E+09	5.91E+07
Peor	842008.8998	2.73E+08	2.73E+16	4.38E+09	8.54E+07
Media	768838.0733	1.96E+07	6.50E+15	1.16E+09	5.96E+07
Desv. Est.	18734.31536	2.57E+07	3.82E+15	4.16E+08	3434168.862
Quality	f11	f12	f13	f14	f15
Mejor	1.17E+09	4.96E+09	1.10E+08	1.24E+09	90128.81782
Mediana	1.17E+09	4.96E+09	1.10E+08	1.24E+09	90128.81782
Peor	6.50E+10	2.77E+11	4.32E+08	1.10E+10	1617640.998
Media	1.88E+09	2.11E+10	1.20E+08	1.48E+09	166518.1205
Desv. Est.	6.39E+09	4.53E+10	3.89E+07	1.25E+09	211142.7828

Tabla resumen de los resultados obtenidos al aplicar 10000 generaciones a los 15 problemas propuestos en [4]

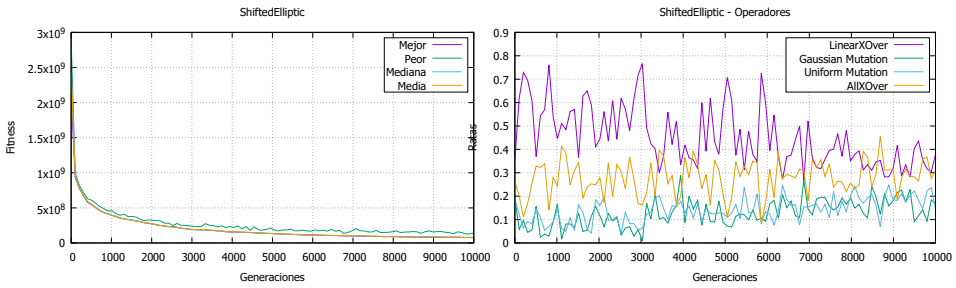

Fig. 16. Resultados para 10000 Iteraciones de la funcion f1

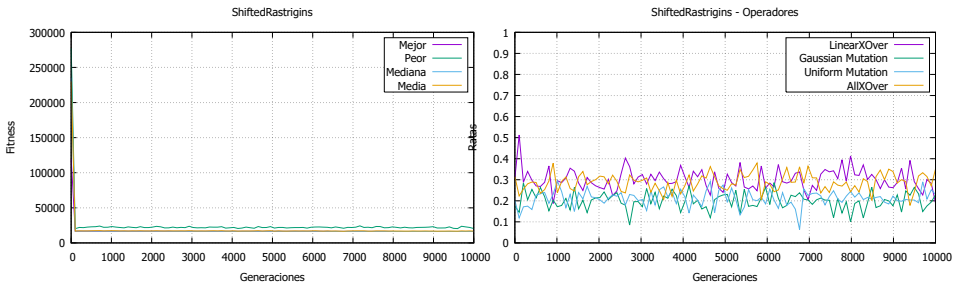

Fig. 17. Resultados para 10000 Iteraciones de la funcion f2

6 Conclusiones

- Se encuentra que el operador que mas domina en el proceso evolutivo es el operador de combinación lineal.
- Para solucionar problemas de temprana convergencia se hizo necesario aplicar un operador de intensidad a la mutación.
- En el proceso de experimentación se encontró que los operadores auto adaptativos presentan un mejor rendimiento que aplicar un algoritmo genético convencional

Trabajo Futuro

Se recomienda la construcción de operadores mas robustos de mutación, puesto que los que se han propuesto en este documento fueron fuertemente dominados por un operador de combinación.

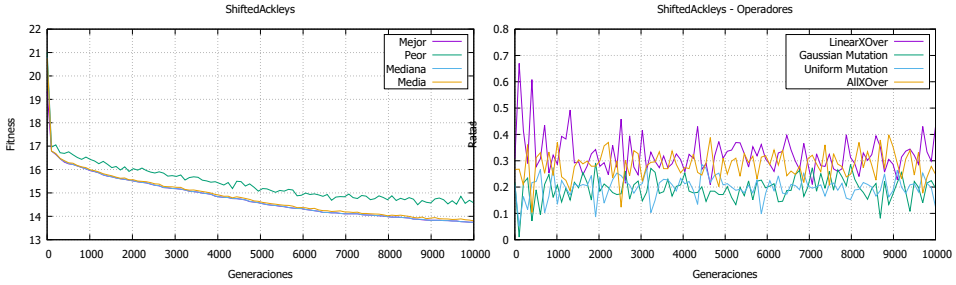

Fig. 18. Resultados para 10000 Iteraciones de la funcion f3

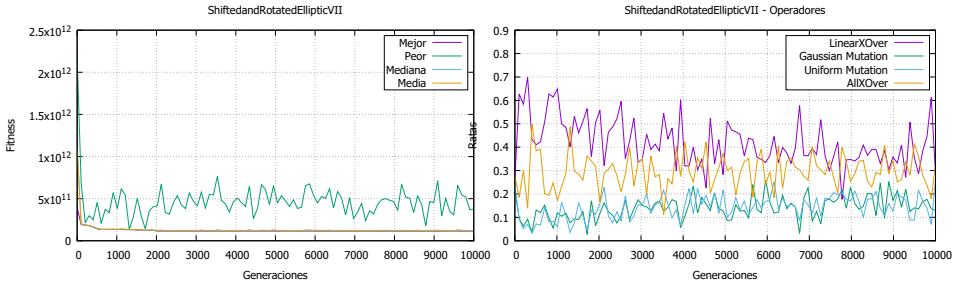

Fig. 19. Resultados para 10000 Iteraciones de la funcion f4

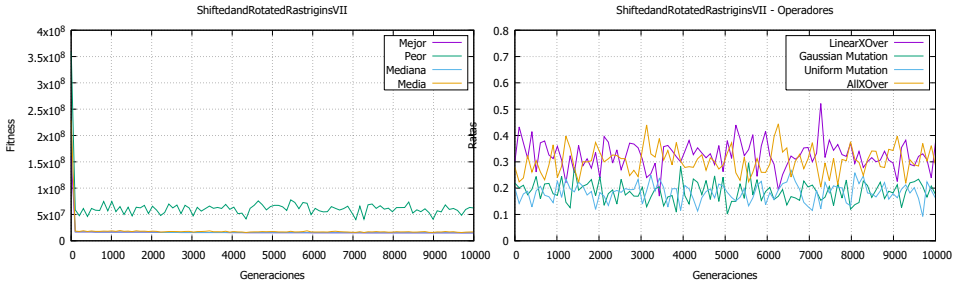

Fig. 20. Resultados para 10000 Iteraciones de la funcion f5

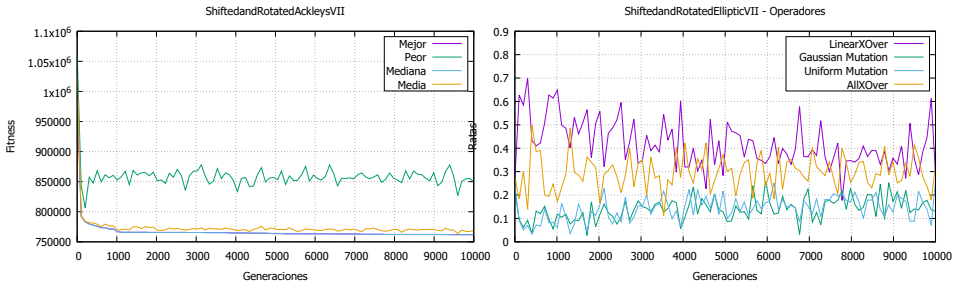

Fig. 21. Resultados para 10000 Iteraciones de la funcion f6

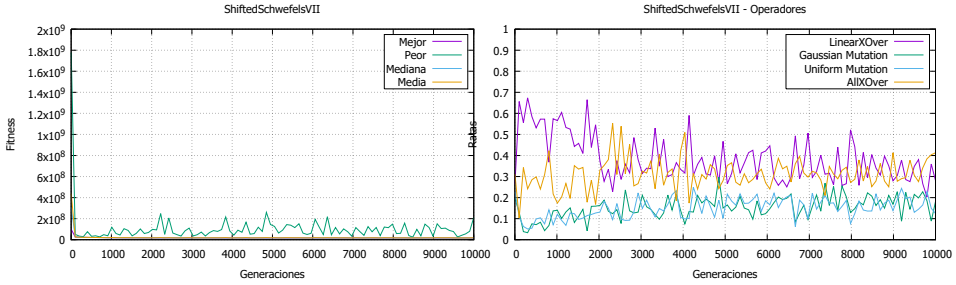

Fig. 22. Resultados para 10000 Iteraciones de la funcion f7

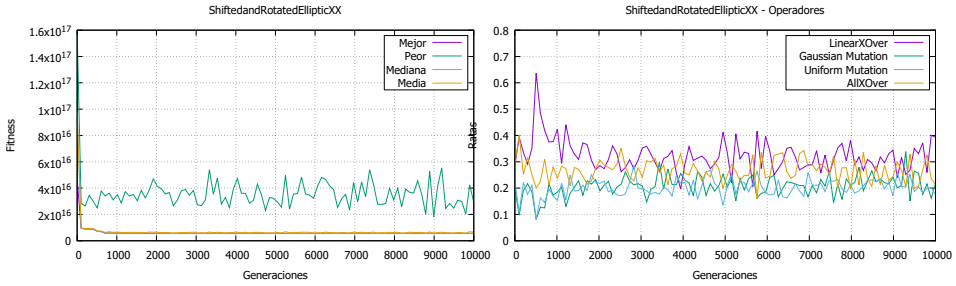

Fig. 23. Resultados para 10000 Iteraciones de la funcion f8

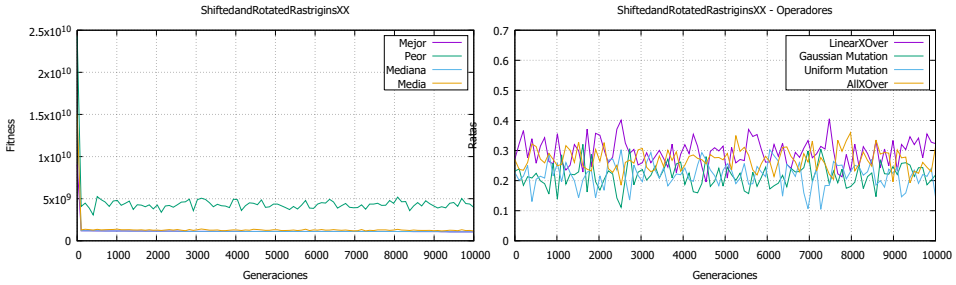

Fig. 24. Resultados para 1000 Iteraciones de la funcion f9

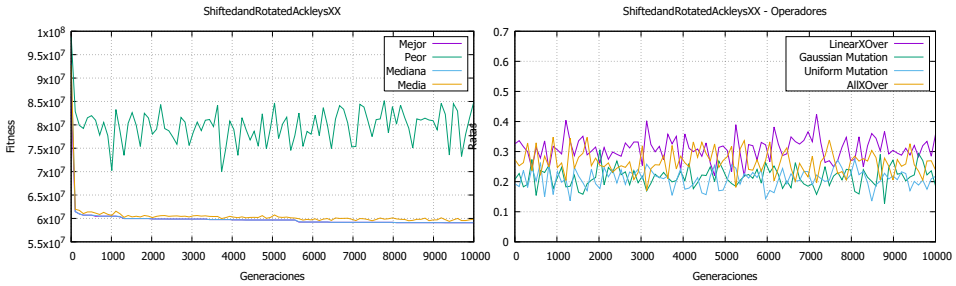

Fig. 25. Resultados para 1000 Iteraciones de la funcion f10

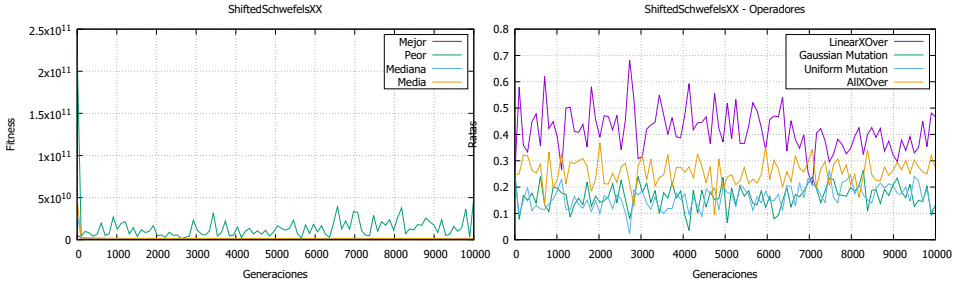

Fig. 26. Resultados para 10000 Iteraciones de la funcion f11

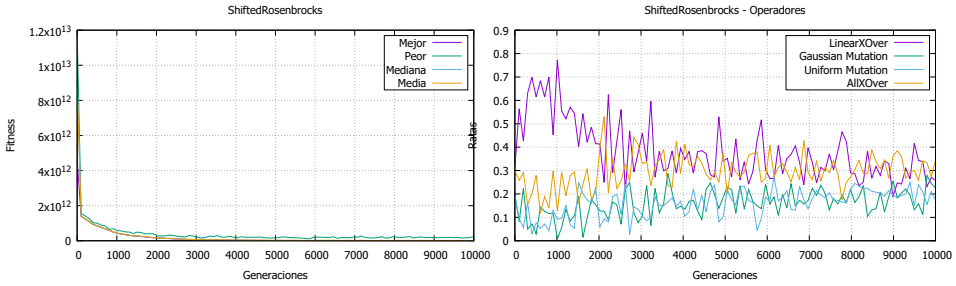

Fig. 27. Resultados para 10000 Iteraciones de la funcion f12

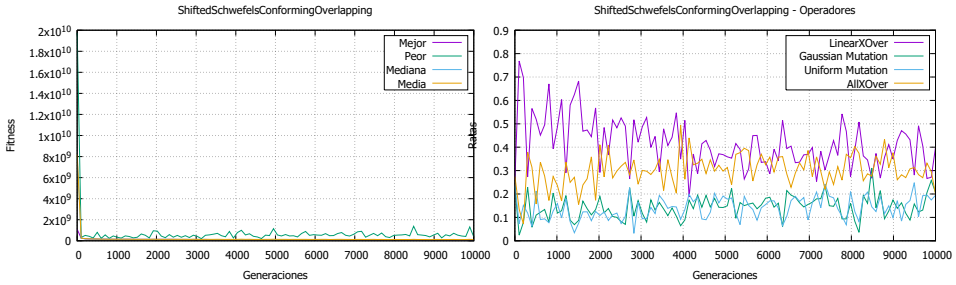

Fig. 28. Resultados para 10000 Iteraciones de la funcion f13

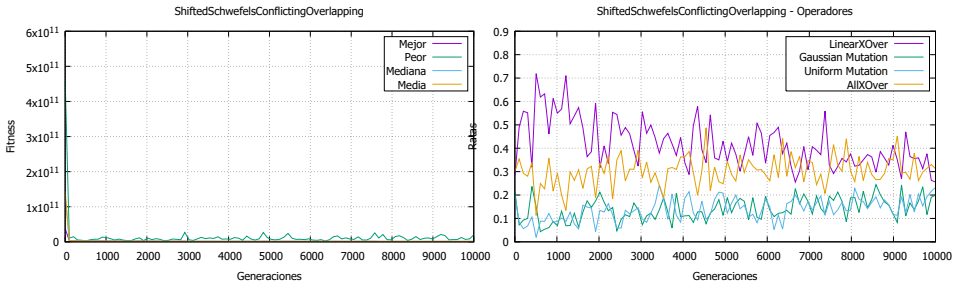

Fig. 29. Resultados para 10000 Iteraciones de la funcion f14

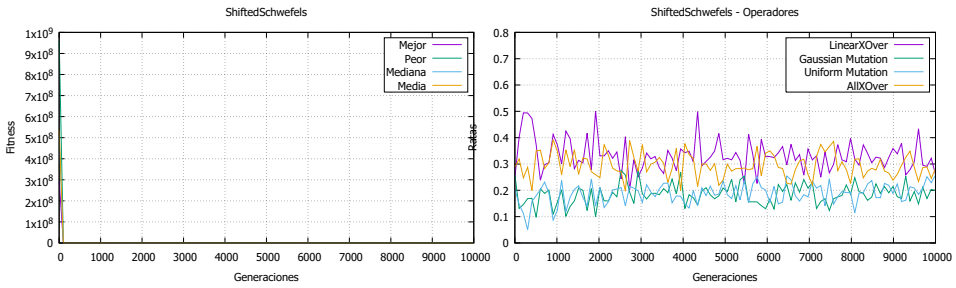

Fig. 30. Resultados para 10000 Iteraciones de la funcion f15

References

1. Anne Auger, *Benchmarking the (1+ 1) evolution strategy with one-fifth success rule on the bbob-2009 function testbed*, Proceedings of the 11th Annual Conference Companion on Genetic and Evolutionary Computation Conference: Late Breaking Papers, ACM, 2009, pp. 2447–2452.
2. Hans-Georg Beyer and Hans-Paul Schwefel, *Evolution strategies–a comprehensive introduction*, Natural computing **1** (2002), no. 1, 3–52.
3. Jonatan Gomez, *Self adaptation of operator rates in evolutionary algorithms*, Genetic and Evolutionary Computation–GECCO 2004, Springer, 2004, pp. 1162–1173.
4. Xiaodong Li, Ke Tang, Mohammad N Omidvar, Zhenyu Yang, Kai Qin, and Hefei China, *Benchmark functions for the cec 2013 special session and competition on large-scale global optimization*, gene **7** (2013), no. 33, 8.
5. Zhenyu Yang, Ke Tang, and Xin Yao, *Large scale evolutionary optimization using cooperative coevolution*, Information Sciences **178** (2008), no. 15, 2985–2999.