Técnicas para contrarrestar la pérdida de variabilidad en un UMDA

Techniques to counter the loss of variability in UMDA

Alfredo Mendoza*

Eunice Ponce de León Sentí**

Elva Díaz Díaz***

Fecha de recepción: 3 de febrero de 2013

Fecha de aceptación: 30 de abril de 2013

Resumen

Debido a la naturaleza de su proceso de aprendizaje, los algoritmos de estimación de la distribución (conocidos como EDA) han padecido, desde sus primeras implementaciones, de problemas de variabilidad (respecto a las soluciones utilizadas en su proceso de optimización); lo que conlleva a estancamientos en óptimos locales o lentitud para encontrar la solución óptima. Diversos métodos y técnicas han sido implementados en estos algoritmos que reducen, y en algunos casos anulan, dicho problema. El UMDA (Univariate Marginal Distribution Algorithm), es la implementación más simple de un EDA; al asumir independencia entre sus variables, simplifica la estimación de la distribución de las soluciones en el producto de las probabilidades marginales. La aleatoriedad de la primera población y el número de individuos utilizados para estimar la distribución son factores que influyen en la variabilidad del algoritmo. Este artículo presenta un análisis comparativo entre los métodos que tratan este problema, involucrando cuatro de los más utilizados en la literatura y dos más propuestos por los autores. También se presenta un análisis de la influencia en la variabilidad de un UMDA de los parámetros utilizados en los operadores de selección y remplazamiento.

*

**

***

Palabras clave

Algoritmos de estimación de la distribución, optimización.

Abstract

Keywords

1. Introducción

Sin importar de qué trate la metaheurística, todas deben explorar de manera efectiva y eficiente el espacio de búsqueda de soluciones del problema por resolver [1]. Los algoritmos evolutivos (EA por su traducción al inglés *evolutionary algorithms*) como son Algorirmos Genéticos, Evolución Diferencial, Algoritmos de Estimación de la Distribución, por mencionar algunos, manipulan un conjunto o población de soluciones a la vez, la cual, para un buen funcionamiento del algoritmo, debe ser lo más diversa posible [2]. La búsqueda de soluciones, durante la evolución del EA es guiada de acuerdo con las características que van teniendo los mejores individuos en cada generación, dicho proceso es conocido como intensificación [1].

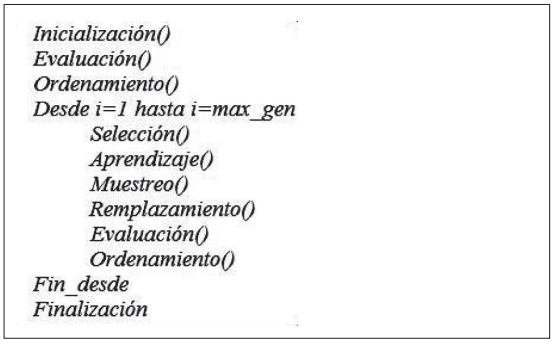
Ya en 1975 John Holland mencionaba que al explotar el espacio de las mejores soluciones se reduce el espacio de búsqueda [3]. Esto es, un EA debe realizar una búsqueda que explote aquellas áreas ya conocidas e incluya los lugares no explorados; por ejemplo, reducir el espacio de búsqueda de manera inteligente. Entre mejor se cubre el espacio de soluciones, se dice que mejor es la variabilidad del algoritmo; en consecuencia, al fenómeno de reducción del espacio de búsqueda se le conoce como pérdida de variabilidad del algoritmo. Al diseñar un AE, se deben cubrir entonces dos requerimientos, la mayoría de las veces en conflicto: la variabilidad y la intensificación. Stutzle [4] menciona: "Una metaheurística cumplirá mejor su propósito en un problema de optimización, si puede proveer un balance entre la explotación de la experiencia de búsqueda acumulada, y la exploración del espacio de búsqueda para identificar nuevas regiones con mejores soluciones".

En el caso de los Algoritmos de Estimación de la Distribución, existe una variante de ellos la cual, por su simplicidad, son más vulnerables a la pérdida de variabilidad, los llamados UMDA. Para el tratamiento de dicho problema, esta investigación tiene las siguientes aportaciones:

- Un compendio de los principales métodos utilizados para contrarrestar la perdida de variabilidad en un UMDA.
- Dos nuevos métodos para contrarrestar la pérdida de variabilidad en un UMDA.
- Un análisis del impacto que tienen en la variabilidad los porcentajes de selección y remplazamiento.
- Un análisis comparativo de los métodos de corrección de la pérdida de variabilidad.

El contenido del artículo es presentado de la siguiente manera: En la parte I, se presentan los conceptos básicos de los Algoritmos de Estimación de la Distribución, incluyendo su teoría general, sus operadores y algunas de sus variantes; en la parte II, se explican las características principales de un UMDA. La parte III comprende aspectos generales del problema de la pérdida de variabilidad y

Figura 1: Proceso de un EDA



Fuente: elaboración propia.

sus métodos de corrección; en la parte IV, se analiza la influencia que tienen los porcentajes de selección y remplazamiento con la variabilidad del algoritmo. Un análisis comparativo entre los diferentes métodos que contrarrestan la pérdida de variabilidad, es presentado en la parte V. Finalmente, las conclusiones son presentadas en el apartado VI (figura 1).

Remplazamiento. Mediante la combinación de la población anterior y los nuevos individuos, se genera una nueva población. Finalización. El algoritmo devuelve como solución, la más reciente aproximación al óptimo de la función.

2. Algoritmos de estimación de la distribución

Los EDA trabajan con una población de individuos del mismo modo que lo hacen los Algoritmos Genéticos (GA, por su traducción del inglés *Genetic Algorithms*), explorando las mejores regiones del espacio de búsqueda. Sin embargo, existen ciertas diferencias en el modo en que cada uno de estos algoritmos manipulan las poblaciones de soluciones: A diferencia de los GA los EDA no tienen operadores de cruzamiento y mutación. Para generar nuevas soluciones, los EDA estiman la distribución conjunta de la población (o de una parte de ella) en un modelo, el cual posteriormente es muestreado [5].

De manera general, un EDA sigue 8 operadores básicos en su proceso de optimización: inicialización, evaluación, ordenamiento, selección, aprendizaje, muestreo, remplazamiento y finalización. El proceso de optimización mediante EDA comienza generando N soluciones (individuos) aleatorias, la adaptabilidad es calculada para cada uno; después de que todos los individuos en la población son ordenados de acuerdo con su adaptabilidad, se selecciona un subconjunto de K individuos, con jKj= (N) y =[0,1]. Posteriormente, se modela la distribución de este subconjunto de individuos y se crean nuevos individuos mediante el muestreo del modelo. El conjunto de padres y el conjunto de descendientes son unidos para crear una nueva población. El proceso anterior se repite hasta que cierta condición de paro es alcanzada; cuando esto ocurre, el algoritmo devuelve su mejor aproximación como solución del problema de optimización. El proceso genérico de los EDA (figura 1), involucra un conjunto de actividades bien definidas para cada uno de los operadores las cuales se describen a continuación:

- Inicialización. El primer paso del proceso de optimización consiste en generar, normalmente de manera aleatoria, una población de soluciones.
- Evaluación. En este operador, se calcula el valor de cada objetivo de la función a optimizar, para cada individuo.
- Ordenamiento. Se ordenan las soluciones considerando su adaptabilidad.
- Selección. Se selecciona un subconjunto de individuos de la población.
- Aprendizaje. Mediante la extracción de información relevante, se modela la distribución de las soluciones seleccionadas.
- Muestreo. Se generan nuevos individuos mediante el muestreo del modelo construido en el operador de aprendizaje.
- Finalización. El algoritmo devuelve como solución, la más reciente aproximación al óptimo de la función.

2.1 Clasificación de los EDA

Siguiendo el criterio de Larrañaga y Lozano [6] y otro realizado por Pelikan [5], los EDA pueden clasificarse siguiendo dos criterios generales: según el dominio de las variables y según la complejidad de la relación entre las variables. Clasificación según el dominio de las variables. Agrupa a los EDA en tres categorías: discretos, continuos y mixtos. Clasificación según la complejidad de la relación entre las variables. Agrupa los EDA en tres categorías: univariados, bivariados y multivariados.

2.1.1 EDA univariados

Uno de los primeros enfoques y el más sencillo que han tenido los EDA es considerar que todas las variables del problema por resolver son independientes. Es decir, la probabilidad de cada variable no depende de ninguna otra, por lo que simplemente se calcula con su probabilidad marginal. La probabilidad conjunta de su distribución se calcula mediante el producto de sus márgenes, como se ilustra en la ecuación 1. Algunos EDA univariados reportados en la literatura son el UMDA, el PBIL y el cGA.

2.1.2 EDA bivariados

Un enfoque más complejo es considerar las relaciones de las variables mediante pares; por ejemplo, la probabilidad condicional de una variable puede depender a lo mucho de otra variable. Estos algoritmos utilizan la población de individuos para calcular información mutua entre todos los pares de variables (Pelikan). De este modo, el proceso de aprendizaje del modelo deja de ser únicamente paramétrico (como el de los univariados) y se extiende a la estructura, i. e. a las relaciones entre las variables. Ejemplos de EDA bivariados son: MIMIC y COMIT y BMDA.

2.1.3 EDA multivariados

Como menciona Larrañaga, algunos enfoques de EDA presentados en la literatura, requieren considerar relaciones de más de dos variables. Los EDA multivariados, de esta manera, estiman la probabilidad conjunta de la población considerando las probabilidades condicionales, en las cuales cada variable puede depender de más de una. EDA multivariados reportados en la literatura son: EcGA, FDA, BOA, etc.

Figura 2. Ejemplo de selección en un UMDA

	X_1	X_2	X_3	X_4	X_5	X_6
	0	1	1	0	1	1
	1	0	0	0	0	0
	0	0	0	0	1	1
	1	1	1	1	1	1
	0	0	1	0	1	1
	0	1	0	1	0	0
$\delta(X)$	**2**	**3**	**4**	**2**	**4**	**4**
$\rho(Xm1)$	**.99**	**.5**	**66**	**33**	**66**	**66**

Fuente: elaboración propia.

2.2 UMDA

Como se mencionó anteriormente, el enfoque más simple que se utiliza en un EDA es considerar independencia entre las variables. El Univariate Marginal Distribution Algorithm (UMDA), introducido por Muhlenbein [7], estima el modelo de la distribución de la población, calculando la probabilidad conjunta de una selección de los mejores individuos en cada generación. Así, la probabilidad de un individuo x en la generación g ($p_g(x)$) considerando que este se encuentra dentro de la selección, se calcula mediante la ecuación 1:

$$p_g(x) = \prod_{i=1}^{n} p_g(x_i) \qquad (1)$$

Donde n es el número de variables. Asimismo, la probabilidad para que cada variable X_i tenga un valor x_i se calcula de la siguiente manera:

$$p_g(X_i = x_i) = \frac{\sum_{j=1}^{N} \delta_j(X_i = x_i)}{N} \qquad (2)$$

Donde N es el número de individuos dentro de la selección y $_j(X_i = x_i)$ se define como:

$$\delta_j(X_i = x_i) = \begin{cases} 1 & \text{si en el j-esimo caso } X_i = x_i \\ 0 & \text{en otro caso} \end{cases} \quad (3)$$

Siguiendo el proceso genérico mostrado en la figura 1, y suponiendo que se desea obtener la mayor cantidad de 1's posible, el UMDA aprender la distribución que siguen los individuos que mejor cumplen dicho propósito; para esto, supongamos que de una población inicial P_0 se seleccionaron 6 de los mejores individuos mostrados en la figura 2. De esta selección se obtienen sus frecuencias marginales ($_j(X_i = 1)$) y posteriormente se calcula $p_g(X_i = 1)$ obteniendo los siguientes resultados:

$$p_g(X_1 = 1) = .33 \qquad p_g(X_2 = 1) = .5$$
$$p_g(X_3 = 1) = .66 \qquad p_g(X_4 = 1) = .33$$
$$p_g(X_5 = 1) = .66 \qquad p_g(X_6 = 1) = .66$$

De la ecuación 1, obtenemos la probabilidad conjunta:

$$p_g(x) = \prod_{i=1}^{n} p_g(x_i) = .01565 \quad (4)$$

Haciendo un muestreo para cada variable según su distribución, el UMDA genera nuevos descendientes. Los descendientes y sus padres conformarán la nueva población que el algoritmo utilizar en la generación g + 1. Siguiendo generación tras generación este proceso, el UMDA se va acercando al óptimo, que en este caso se logra cuando x = f1g[6].

FiguFigura 3. Evolución de la probabilidad

Generación	P(X1)	P(X2)	P(X3)	P(X4)	P(X5)	P(X6)
1	0.6000	0.6200	0.6600	0.6000	0.7000	0.7200
2	0.7800	0.8000	0.6400	0.7600	0.8400	0.8600
3	0.9200	0.8600	0.7000	0.9000	0.9600	0.9600
4	0.9600	0.9400	0.9400	0.9200	0.9800	1.0000

Fuente: elaboración propia.

Figura 4. Ejemplo de pérdida de variabilidad

	X_1	X_2	X_3	X_4	X_5	X_6
	0	1	1	0	1	1
	0	0	1	0	0	1
	0	0	0	0	1	1
	0	1	1	1	1	1
	0	0	1	0	1	1
	0	1	0	1	0	1
$\delta(Xi)$	0	3	4	2	4	6
$p(X_i=1)$	0	.5	.66	.33	.66	1

Fuente: elaboración propia.

3. Pérdida de variabilidad en un UMDA

Conforme evoluciona el UMDA, las probabilidades marginales cambian. La figura 3 muestra dicho cambio en las primeras tres generaciones. Como se señaló anteriormente la población inicial es aleatoria, por lo que en ella podemos encontrar alguno de los casos mostrados en la figura 4, donde en algunas de las variables, la probabilidad es 1 o 0. Tener estos valores extremos en la población inicial puede llevar al UMDA a estancarse en un óptimo local, sobre todo cuando se manejan poblaciones pequeñas, i. e. poblaciones en las que el número de variables es mayor al número de individuos que componen la población; esta condición puede evitarse asegurando que la población inicial no contenga casos absolutos [8]. Sin embargo, si se desea mantener la condición de aleatoriedad en la población inicial, es necesario utilizar otros métodos, más aún, este fenómeno conocido como pérdida de variabilidad, ocurre ciertamente en la población inicial, pero puede también ocurrir a lo largo de la evolución del algoritmo [8].

Para disminuir, y en el mejor de los casos eliminar, la pérdida de variabilidad en un UMDA, se han desarrollado diversos méto-

dos; en la siguiente sección se explican tres de ellos y se proponen dos más, con algunas de sus variantes.

3.1 Métodos de corrección de variabilidad

El problema de pérdida de variabilidad en un UMDA ha sido ampliamente estudiado, algunos métodos se enfocan en no permitir que se generen individuos que lleven las probabilidades a algunos de los extremos, otros limitan el rango de probabilidades y otros modifican la forma en que se obtiene la probabilidad de cada variable. Los métodos más comunes son: correción de Lapalace (con sus variantes) y la Corrección por límites. En esta sección se explica cada uno (el lector puede referirse a la bibliografía para conocer los fundamentos teóricos de cada método); mientras que en la sección 4 se realizar un análisis comparativo entre ellos.

3.1.1 Corrección de Laplace Básica

La correción de Laplace es un método que, al ser utilizado por un UMDA, permite lidiar con casos donde la probabilidad de una variable es cero. Esto ocurre cuando la probabilidad de que una variable X_j pueda llegar a tener un valor x_i ($p(X_i = x_j)$, lo que puede llegar a estancar al algoritmo en un optimo local. Para evitar estos casos, el método de corrección de Laplace agrega un individuo que contiene los valores de las variables que no aparecieron en la selección. Existen dos maneras de aplicar este método en un UMDA discreto, donde una variable X_j = [0; 1]: agregando los casos absolutos 0 y 1 a la selección; o fijando el valor de las variables X_j que presentan la probabilidad absoluta (0 o 1) en el valor que no aparece en la selección y generando aleatoriamente los valores de las demás variables del individuo a agregar. Para cada caso se agregan 2 y 1 individuos respectivamente, el diseñador del algoritmo decidirá si continúa con una selección de tamaño N + i (con i igual al número de individuos agregados) o elimina i indiviuos (posiblemente seleccionados al azar) de ella.

3.1.2 Corrección de Laplace

Este método consiste en utilizar una probabilidad apriori, tal y como se utiliza en otros métodos de estimación como las redes bayesianas, fundamentado en la distribución de probabilidad. Para implementarlo en un UMDA, la probabilidad de X_i se calcula reemplazando (2) con la fórmula presentada en la ecuación 5. Donde el valor de indica que tanto incluye la probabilidad apriori en la probabilidad actual (2 [0; 1]) y es el porcentaje de selección. Una desventaja de este método es que el algoritmo tarda más en encontrar el óptimo que utilizando otras correcciones.

$$p(x_i) = \frac{\sum_{i=1}^{\tau N} x_i + \alpha}{\tau N + 2\alpha} \quad (5)$$

3.1.3 Corrección de Laplace Incremental

Con la misma idea básica que el método anterior, la corrección incremental de Laplace, considera una probabilidad apriori, si diferencia radica en que la probabilidad apriori se va actualizando conforme van avanzando las generaciones. De esta manera, el cálculo de la probabilidad de x_i se real-iza siguiendo la ecuación 6.

$$p_g(x_i) = \frac{\sum_{i=1}^{N} x_i + 2\alpha p_{g-1}(x_i)}{\tau N + 2\alpha} \quad (6)$$

3.1.4 Corrección con límites

El célculo de la probabilidad con este método se realiza utilizando la ecuación 7. Branke [8] menciona que este método no interfiere con la distribución de probabilidad, como lo hacen los métodos de Laplace; la correción con límites simplemente previene que $p(x_i)$ llegue a ser 0 o 1. Los creadores de este método, recomiendan que el valor de sea 1=L donde L es el número de variables del problema.

$$p(x_i) = \begin{cases} \beta & \text{si } p(x_i) < \beta \\ 1-\beta & \text{si } p(x_i) > 1-\beta \\ p(x_i) & \text{en otro caso} \end{cases} \quad (7)$$

3.1.4 *Corrección con nivel de confianza*

En algunas estimaciones estadísticas comúnmente se utiliza un nivel de confianza (1); el método de Corrección con Nivel de Confianza para un UMDA, define la confianza que se le tendrá a la estimación de la probabilidad de x_i mediante el parámetro (= [0; 1]). La figura 10 muestra gráficamente esta idea. El valor del paáametro para cada extremo del intervalo se define como =2; de esta manera, la $p(x_i)$ se calcula como lo indica la ecuación 8. Como se demuestra en las pruebas, los valores grandes aumentan la variabilidad del algoritmo pero retrasan (y en algunos casos impiden) alcanzar el óptimo de la función.

$$p(x_i) = \begin{cases} 1-(\alpha/2) & \text{si } p(x_i)=1 \\ 0+(\alpha/2) & \text{si } p(x_i)=0 \end{cases} \quad (8)$$

3.1.6 *Corrección con aproximación*

La búsqueda que realiza un AE, como se mencionó anteriormente y que da origen al problema tratado en el artículo, es guiada por las mejores soluciones presentes en la población; sin embargo se debe mantener la línea de búsqueda para llevar al algoritmo a encontrar la solución óptima (cuando es conocida) o a la mejor aproximación (cuando es desconocida). Siguiendo el enfoque del método anterior, el método de Corrección con Aproximación utiliza el mismo operador el cual no mantiene un valor fijo, sino uno que va disminuyendo conforme evoluciona el algoritmo; así se mantiene la variabilidad, pero se sigue con la línea de búsqueda.

La probabilidad de x_i está dada por:

$$p(x_i) = \begin{cases} 1-s_g & \text{si } p(x_i)=1 \\ 0+s_g & \text{si } p(x_i)=0 \end{cases} \quad (9)$$

$$\text{Con : } s_g = d\left(1-\left(\frac{1}{max\_gen} * g\right)\right)$$

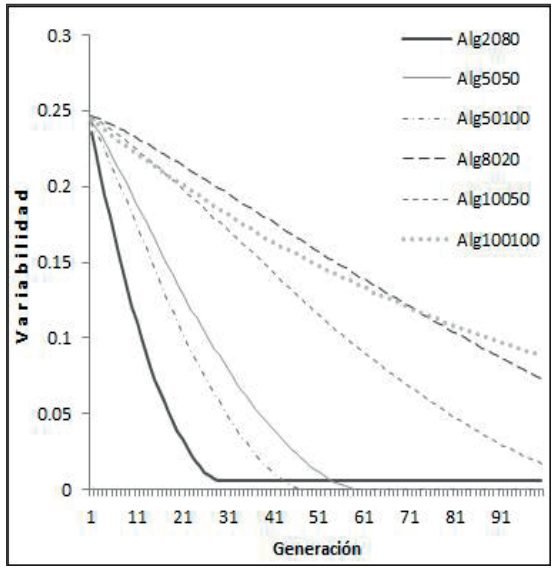
Donde g es la generación actual, maxgen es el número máximo de generaciones y *d* es un parámetro que indica el valor del incremento al valor de s_g. El valor *d* hace que el valor de s_g vaya disminuyendo conforme avanzan las generaciones de manera que el valor de s_g es *d* en la generación 1 y 0 en la generación maxgen. El concepto fundamental de este método es ayudar al algoritmo en las primeras generaciones, evitando probabilidades absolitas (0 o 1) y disminuir dicha influencia conforme el algoritmo evoluciona.

3.2 Relación de la variabilidad y los porcentajes de selección y reemplazamiento

Dos de los parámetros iniciales de un UMDA (y en general de un EDA), son los porcentajes de selección y de reemplazamiento. El primero, indica la proporción de la población sobre la cual se realizar el análisis estadístico, mediante el cual se estima el modelo de la distribución (figura 5). Mientras tanto, el porcentaje de reemplazamiento indica la proporción de la población que será reemplazada por los nuevos individuos creados a partir del muestreo del modelo estimado.

La estrategia que un EDA sigue, en este sentido, es muy variada, dentro de las más

Figura 5. Variabilidad con porcentajes de selección y reemplazamiento



Fuente: elaboración propia.

comunes se encuentra la llamada *truncation selection*, la cual consiste en seleccionar el 50 % de la población para estimar el modelo, y formar una nueva población con estos individuos más otro 50 % de nuevos individuos muestreados del modelo. Su nombre lo recibe por el hecho de cortar o truncar la población original a la mitad y a partir de ella generar una nueva. Sin embargo, la variación de los porcentajes genera diferentes estrategias particulares. En este apartado, analizaremos cómo la variación de dichos porcentajes puede llegar a afectar la variabilidad del algoritmo.

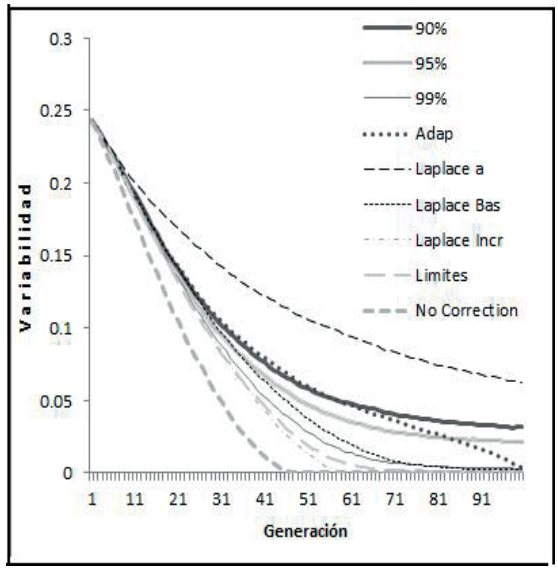
Según la intuición del diseñador del algoritmo y de su expertise, se pueden fijar diferentes valores en los porcentajes de selección y reemplazamiento; en este artículo se eligieron seis combinaciones, las cuales intentan cubrir diferentes (y extremas) tendencias en los valores.

Estrategia 1: 20 % selección, 80 % reemplazamiento estrategia 2: 50 % selección, 50 % reemplazamiento estrategia 3: 50 % selección, 100 % reemplazamiento estrategia 4: 80 % selección, 20 % reemplazamiento estrategia 5: 100 % selección, 50 % reemplazamiento estrategia 6: 100 % selección, 100 % reemplazamiento

Las pruebas se realizaron sin utilizar ninguno de los métodos de corrección mencionados en el apartado anterior; los resultados mostrados en la gráfica XX únicamente indican el nivel de variabilidad que se obtuvo con las diferentes estrategias de selección y reemplazamiento.

Con los resultados mostrados en la figura 6 se puede ver que la variabilidad que tenga el algoritmo puede ser alterada por los diferentes valores que se le asignen a los porcentajes de selección y reemplazamiento. Dentro de la comparación realizada en el siguiente apartado, se incorporarán combinaciones de estos porcentajes con algunas de las técnicas de corrección. Para una revisión global de los resultados, el lector puede referirse a la sección de conclusiones.

Figura 6. Variabilidad con diferentes técnicas de corrección



Fuente: elaboración propia.

4. Comparación entre técnicas de corrección

En esta sección se comparan los diferentes métodos que tratan el problema de la pérdida de variabilidad, así como las combinaciones de porcentajes de selección y reemplazamiento que mejores resultados (en cuanto a variabilidad) presentaron, aplicados a un UMDA.

Los algoritmos fueron aplicados en un problema one-max de 500 variables.

One-max. También conocido con el nombre de BitCount-ing, el problema One-max consiste en maximizar el número de unos en una cadena binaria [9]. De manera formal, el problema se define como:

$$max\ F(x) = \sum_{i=1}^{N} x_i$$

Donde N es el número de variables. De esta manera, el caso particular del problema por resolver en la prueba es:

$$max\ F(x) = \sum_{i=1}^{500} x_i$$

Los parámetros iniciales del algoritmo son:

Población: 100 individuos

Generaciones máximas: 100

Corridas independientes: 20

La variabilidad se calculó como el promedio de la varianza sobre todas las variables [8]. La primera prueba se realizó para algoritmos con 50 % de selección y 50 % de reemplazamiento. La figura 4 muestra la variabilidad presentada a lo largo de las 100 generaciones. En ella se presentan los valores para los diferentes métodos de corrección.

Como se puede observar (figura 7), el método de corrección de Laplace es el que mejor adaptabilidad agrega al algoritmo, seguido por el método del 90 % de nivel de confianza, el de aproximación y el de 95 % de nivel de confianza. Los niveles más bajos de variabilidad fueron presentados por el algoritmo sin corrección, el método de Laplace Incremental y el método de límites.

Como se mencionó en un principio, el exceso de variabilidad le puede dificultar al algoritmo la búsqueda de la solución óptima; para este hecho, la figura 5 complementa los resultados de la figura anterior, presentando el número de generación en el cual el algoritmo encontró por primera vez al menos una solución óptima, que para este caso, es 500.

La segunda prueba se realizó combinando los métodos de corrección que mejores resultados obtuvieron en la primera prueba, con los porcentajes de selección y reemplazamiento que mejores niveles de variabilidad presentaron. Los resultados se muestran en la figura 8. La variación de los porcentajes de selección y reemplazamiento agregan estabilidad a los niveles de variabilidad durante las generaciones; sin embargo, en ninguno de los algoritmos se encontró la solución óptima para el problema one-max de 500 variables.

5. Conclusiones

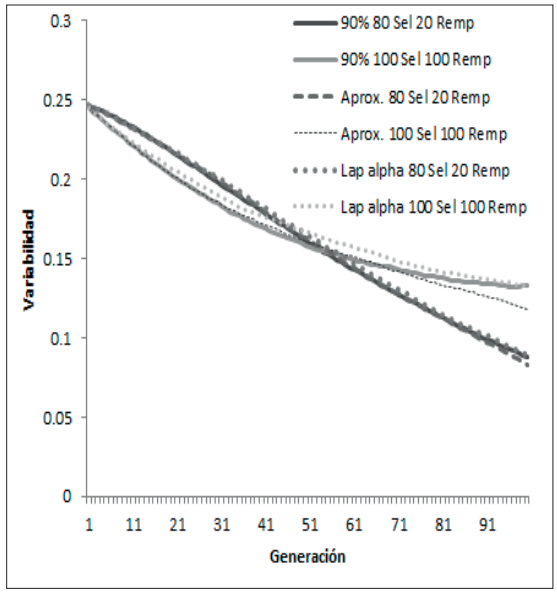
En el presente artículo se analizaron diferentes formas de contrarrestar la pérdida de variabilidad que comúnmente sufre un UMDA, y en general los EDA, al utilizar

Figura 7. Generación vs. Óptimo

Método	Generación
90%	No se encontró óptimo
95%	No se encontró óptimo
99%	82
Aproximación	99
Laplace alpha	87
Laplace Básica	No se encontró óptimo
Laplace Incremental	No se encontró óptimo
Limites	74
Sin Corrección	No se encontró óptimo

Fuente: elaboración propia.

Figura 8: Porcentajes de Selección y Remplazamiento + Corrección



Fuente: elaboración propia.

poblaciones pequeñas a comparación con el número de variables del problema que resuelven. Se compararon seis métodos de corrección, los cuales agregan diferentes niveles de variabilidad al algoritmo en el que se aplican. Si se compara a la par, la variabilidad con la eficiencia de búsqueda de la solución óptima, se pudo observar que algunas técnicas ofrecen mejores resultados que otras. En esta investigación también se agregó un análisis del efecto que tienen los porcentajes de selección y reemplazamiento del algoritmo, sobre su nivel de variabilidad. Se observó que algunos valores en estos porcentajes agregan y mantienen buenos niveles de variabilidad durante las generaciones, especialmente cuando se combinan con algún método de corrección.

Es necesario denotar que es responsabilidad del diseñador del algoritmo agregar o disminuir el nivel de variabilidad de su estrategia debido principalmente al hecho analizado aquí, de que altos niveles de variabilidad, pueden dificultar y en consecuencia desviar la ruta hacia la solución óptima. Es común que un EDA decaiga en sus niveles de variabilidad, para estos casos, el diseñador puede utilizar esta investigación como una buena referencia de métodos y técnicas que le ayudarán a mejorar su trabajo, encontrando la combinación adecuada para su caso particular.

6. Referencias

[1] C. Blum and A. Roli, Metaheuristics in Combinatorial Optimization: Overview and Conceptual Comparison, *Computing*, vol. 35, núm. 3. 2003.

[2] M. Mitchell, *An Introduction to Genetic Algorithms*. Cambridge, MA, USA, MIT Press. 1996.

[3] J. H. Holland, Adaptation in Natural and Artificial Systems. Ann Harbor, University of Michigan Press. 1975.

[4] M. Dorigo and T. Stutzle, The Ant Colony Optimization Metaheuristic: Algorithms, Applications, and Advances, Handbook of Metaheuristics International Series in Operations Research & Management Science. 2003.

[5] M. Hauschild and M. Pelikan, A Survey on Estimation of Distribution Algorithms, MEDAL Report No. 2011004, p. 0113. 2011.

[6] P. Larrañaga and J. A. Lozano, Estimation of Distribution Algorithms: A new tool for Evolutionary Computation. 2002.

[7] H. Muhlenbein and G. Paa, From recombination of genes to the estimation of distributions, Parallel Problem Solving from Nature. 1996.

[8] J. L. S. Jurgen Branke, Clemens Lode, Addressing Sampling Errors and Diversity Loss in UMDA, Proceedings of the 9th annual conference on Genetic and evolutionary computation, 2007.

[9] J. Brownlee, Clever Algorithms: Nature-Inspired Pro-gramming Recipes. Jason Brownlee, p. 436. 2011.