OPTIMIZACIÓN MULTIOBJETIVO USANDO UN ALGORITMO GENÉTICO Y UN OPERADOR ELITISTA BASADO EN UN ORDENAMIENTO NO-DOMINADO (NSGA-II).

Multi-Objective Optimization Using Elititst Non-Dominated Sorting Genetic Algorithm (NSGA-II)

RESUMEN
Este artículo presenta conceptos básicos de la optimización multiobjetivo, enfocados hacia la descripción del algoritmo NSGA-II, el cual es conocido en la literatura especializada por su buen desempeño y por los pocos cambios necesarios respecto al algoritmo mono-objetivo para ser implementado apropiadamente.

PALABRAS CLAVES: combinatorial, genético**,** multiobjetivo, optimización

ABSTRACT
This article presents basic concepts of the multiobjective optimization, focused towards the description of the algorithm NSGA-II, which is known in the specialized literature due to its good performance and the few necessary changes with respect to the mono-objective algorithm to be implemented appropriately.

CÉSAR AUGUSTO PEÑUELA MENESES
Ingeniero Electricista.
Estudiante de Maestría del programa de ingeniería eléctrica
Universidad Tecnológica de Pereira
cpenuela@orbitel.net.co

MAURICIO GRANADA ECHEVERRI
Ingeniero Electricista.
Profesor asistente.
Programa de ingeniería eléctrica.
Universidad Tecnológica de Pereira
magra@utp.edu.co

KEYWORDS: *combinatorial, genetic, multiobjective, optimization.*

1. INTRODUCCIÓN

En el proceso de fabricar un producto, o en el simple hecho de mejorar la calidad de vida de algún individuo, se deben tomar decisiones basadas en una gran cantidad de variables con características y comportamientos diferentes y, al evaluarlas adecuadamente, elevar notablemente el beneficio final. Al tratar de encontrar una solución con este enfoque se ingresa inevitablemente en el campo de la optimización matemática, el cual alberga un conjunto bastante amplio de metodologías donde cada opción puede ser tan compleja como el problema mismo.

Normalmente, las metodologías de optimización se concentran en adecuar un conjunto de elementos de manera que se mejore el resultado dado por una función objetivo. Sin embargo los problemas reales involucran otra serie de objetivos que pueden ser de tanto interés como el que se optimizó, e incluso ser tan relevantes y conflictivos que harían inviable la solución obtenida. La optimización multiobjetivo basada en técnicas evolutivas es una meta heurística que surgió con el fin de resolver este tipo de problemas, caracterizada por ser capaz de obtener un conjunto de soluciones, con los mejores compromisos entre los objetivos optimizados (*frente óptimo de pareto*).

En el presente trabajo se pretende explicar el funcionamiento básico de uno de los más conocidos algoritmos multiobjetivos propuesto en la literatura especializada, con el fin de presentar una herramienta matemática que pueda ser usada para resolver problemas complejos de optimización multiobjetivo.

Fecha de Recepción: 30 mayo de 2007
Fecha de Aceptación: 18 Julio de 2007

2. PROBLEMA BÁSICO

En general, un problema de optimización multiobjetivo se formula como:

$$\begin{aligned} &\min/\max \quad f_m(x) && m = 1,2,...,M \\ &s.a. \\ &\quad g_i(x) \geq 0 && i = 1,2,...,I \\ &\quad h_k(x) = 0 && k = 1,2,...,K \\ &\quad x_j^L \leq x_j \leq x_j^U && j = 1,2,...,J \end{aligned} \tag{1}$$

La principal diferencia con un modelo mono-objetivo radica en la presencia de un grupo de funciones que deben ser optimizadas simultáneamente. De esta forma, lo que se busca es encontrar un vector de variables de estado $x=(x_1,x_2,\ldots x_j)$ que cumpla con en el conjunto restricciones y donde las funciones objetivos resultantes sean optimizadas.

El espacio de solución, representado por todas las combinaciones posibles en el valor de las variables, genera un segundo espacio vectorial conocido como *Espacio Objetivo* y denotado por $f_n(x)=z=(z_1,z_2,\ldots z_M)$.

Cuando se trata con problemas de optimización mono-objetivo una alternativa de solución se considera mejor que otra si produce una solución objetivo de menor valor

para el caso de minimización, o de mayor valor para el caso de maximización. Pero en los problemas multiobjetivo este criterio debe ser revaluado, ya que se considera al mismo tiempo funciones de minimizar y funciones de maximizar. Además, el orden de magnitud de cada función puede ser diferente, haciendo inviable una comparación directa.

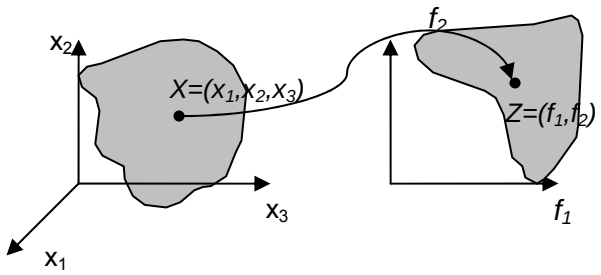


Figura 1. Representación del Espacio de Solución y su correspondiente Espacio Objetivo

Por tanto, los algoritmos multiobjetivo introducen el operador de *Dominancia*, el cual define que una solución x(1) domina otra solución x(2) si se cumplen las siguientes condiciones:

- La solución x(1) no es de menor calidad que x(2) en todos los objetivos.
- La solución x(1) es estrictamente mejor que x(2) en al menos uno de los objetivos.

Aplicando iterativamente estas reglas sobre un conjunto cualquiera de soluciones de un problema de optimización multiobjetivo, se puede establecer cuales son las alternativas dominantes, conocidas como *Conjunto No Dominado*. Las soluciones restantes forman parte del Conjunto de *Soluciones Dominadas*. Si se logra establecer cuál es el conjunto de Soluciones Dominantes a través de todo el espacio objetivo, entonces se habla de *Frente óptimo de Pareto*.

3. MÉTODO DE KUNG PARA LA OBTENCIÓN DEL CONJUNTO NO DOMINADO.

La mayoría de algoritmos de optimización multiobjetivo usan el concepto de dominancia en sus procesos de búsqueda. En estos algoritmos, dos soluciones son comparadas a fin de establecer cual solución domina o no a la otra.

El método discutido por Kung en [1], ver figura 3, propone una división recursiva de la población y es considerado en la literatura especializada como el método computacionalmente más eficiente. El primer paso consiste en ordenar descendentemente la población P según la importancia del valor de la primera función objetivo. Posteriormente, la población es dividida en dos subpoblaciones I (izquierda) y D (derecha) de forma recursiva (ver líneas 5 y 6 del código mostrado en la tabla 2). Lo anterior implica que la subpoblación I es de mejor calidad que la D desde el punto de vista de la primera función objetivo. Así, es posible verificar el criterio de dominancia, respecto a la segunda función objetivo, entre la subpoblación D y la I (el proceso es aplicable para problemas con más de dos funciones objetivo). Las soluciones de D que no son dominadas por cualquier miembro de I son combinadas con los miembros de I para formar una población no dominada M (ver líneas 9 y 16). La conformación de la población M y la verificación de dominancia tienen lugar en el momento en que el tamaño de I y de D sea igual a 1, es decir, hasta que las divisiones recursivas de las subpoblaciones permitan comparar sólo un individuo de la población I con uno de la población D. El diagrama de flujo de la Figura 3 ilustra la discusión anterior considerando la definición de los siguientes parámetros:

V: Matriz que contiene los valores de todas las funciones objetivo para cada uno de los individuos de la población.

P: Vector que contiene el número de cada *individuo* y corresponde a la primera columna de la matriz *Población*.

N: Número de *individuos* de la población.

M: Número de funciones objetivo del problema.

TipoOpt: Vector que define el tipo de optimización (minimización=0 o maximización=1) de cada una de las funciones objetivo. Por ejemplo, TipoOpt=[0, 1] significa que F_{obj1} es de minimización y F_{obj2} es de maximización.

Este algoritmo, además de entregar el conjunto de soluciones dominantes, permite organizar las soluciones en frentes que indican el nivel o rango de dominancia que posee una alternativa frente a las demás. Así por ejemplo, para la población mostrada en la tabla 1, se obtienen los resultados mostrados en la figura 2.

Tabla 1. Población de individuos.

Vector P	Matriz [V]	
Individuo	Fobj1	Fobj2
1	0.2285	11.5128
2	0.3902	7.4299
3	0.7559	7.1822
4	0.1688	28.4664
5	0.457	11.8462
6	0.4078	8.0556
7	0.3727	7.7673
8	0.1195	26.9118
9	0.7348	6.5577
10	0.2813	8.2604
11	0.6785	3.863
12	0.8121	3.5582

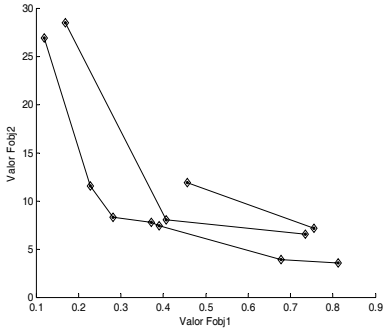


Figura 2. Organización en Frentes de Dominancia de las alternativas de solución.

Si se quieren obtener los demás frentes, correspondientes a los individuos que no hacen parte del conjunto no

dominado, el procedimiento consiste en retirar de la población los individuos que ya han sido ubicados en un frente y repetir el proceso de bisección recursiva.

4. ALGORITMO NSGA-II.

Los algoritmos multiobjetivos requieren para su desarrollo de métodos matemáticos de optimización sobre una población de soluciones, por lo que se ha encontrado en los algoritmos genéticos una propuesta firme, dadas sus características de diversidad y confiabilidad. Sin embargo se debe mantener una mente abierta y posibilitar el uso de otros mecanismos tales como las colonias de hormigas y las partículas swarp. A continuación se dará una revisión general del algoritmo NSGA-II, cuya primera versión se apoya en algoritmos genéticos.

Es clasificado como de tipo elitista, ya que incorpora un mecanismo de preservación de las soluciones dominantes a través de varias generaciones de un algoritmo genético. El proceso se inicia a partir de un conjunto de tamaño N de soluciones (Padres) obtenidas al azar o a través de un constructivo suave. Las siguientes generaciones son determinadas usando mecanismos modificados de selección cruzamiento y mutación definidos por el algoritmo genético clásico.

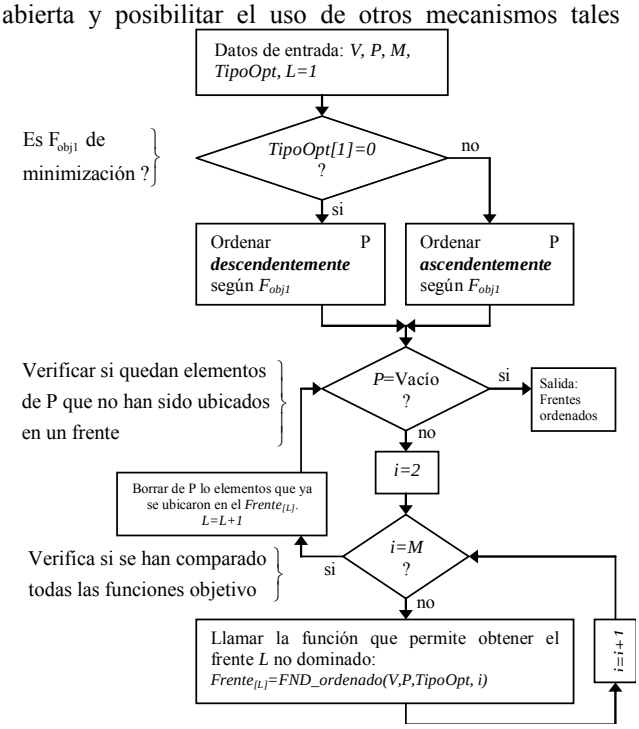


Figura 3. Diagrama de flujo del Método de Bisección Recursiva.

Tabla 2. Seudo código del método de bisección recursiva

Método de Kung (bisección recursiva)

```
1.  function [Frente] = FND_ordenado(V, P,TipoOpt, i)
    % La entrada i corresponde a la i-ésima función objetivo con
    % la que se comparará la población actual.
2.  N=size(P,1);
3.  PosDividir=round((N)/2);
4.  if N>1
     %Se divide el frente y se llama la función recursivamente.
5.    [I]= FND_ordenado (V,P ([1:PosDividir],:), TipoOpt, i);
6.    [D]= FND_ordenado (V,P([PosDividir+1:N],:), TipoOpt, i);
7.    if TipoOpt(i)==0      % Si el problema es de minimización
8.      if V(D(size(D,1)),i)<=V(I(size(I,1)),i)   % Si I domina a D
9.        M=[I;D];
10.     else
11.       M=[I];
12.     end
13.   end
14.   if TipoOpt(i)==1    % Si el problema es de maximización
15.     if V(D(size(D,1)),i)>=V(I(size(I,1)),i) % Si I domina a D
16.       M=[I;D];
17.     else
18.       M=[I];
19.     end
20.   end
21.   Frente=M;
22. else
23.   Frente=P;
24. end
```

4.1 Proceso de Selección, Cruzamiento y Mutación.

Sobre la población actual (Padres) son seleccionadas N parejas de soluciones escogidas aleatoriamente. Cada pareja compite en un torneo donde gana la alternativa que pertenezca al rango de mejor calidad. Si las alternativas en competencia pertenecen al mismo frente, entonces gana la que introduzca un mayor grado de diversidad al conjunto en construcción. Los vencedores de cada torneo son los únicos facultados para obtener descendencia, el cruzamiento y mutación se manejan de igual forma al mostrado por el algoritmo genético clásico. De esta manera, lo que se espera es que la información genética de las alternativas dominantes esté presente en las siguientes generaciones y atraiga al resto de la población hacia sus vecindades.

4.2 Operador de Apilamiento

Los algoritmos multiobjetivo buscan encontrar el mayor número posible de soluciones que pertenezcan al frente de pareto. Por tanto, es necesario que la población se mantenga tan diversa como sea posible. El operador de apilamiento permite cuantificar el espacio alrededor de una alternativa que no se encuentra ocupada por ninguna otra solución. Para esto se debe calcular el perímetro del cuboide formado por las soluciones vecinas que poseen el mismo rango de dominancia que la alternativa i, lo cual se describe por medio de la ecuación (2) como:

$$d_i = \sum_{m=1}^{M} \left| \frac{f_m^{(I_{i+1}^m)} - f_m^{(I_{i-1}^m)}}{f_m^{\max} - f_m^{\min}} \right| \qquad (2)$$

Donde I^m es un vector que indica la alternativa de solución vecina a la alternativa *i*, f_m^{max} y f_m^{min} son los valores máximo y mínimo sobre todo el espacio de solución de la función objetivo m, y M es el número de funciones objetivo optimizadas.

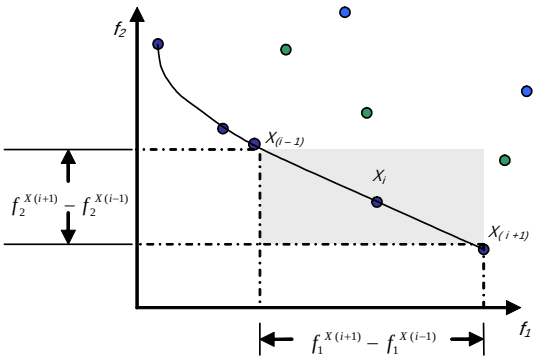


Figura 4. Distancia de apilamiento para la alternativa de solución Xi.

Por ejemplo para el caso de la figura 4, con dos funciones objetivo y tres rangos de dominancia para las 10 alternativas mostradas, el semiperímetro del área sombreada está dado por medio de la ecuación (2) como:

$$di = \left| \frac{f_1^{X(i+1)} - f_1^{X(i-1)}}{f_1^{\max} - f_1^{\min}} \right| + \left| \frac{f_2^{X(i+1)} - f_2^{X(i-1)}}{f_2^{\max} - f_2^{\min}} \right|$$

De esta manera se puede afirmar que para un conjunto de soluciones en competencia, la alternativa que introduce el mayor nivel de diversidad es aquella con la mayor distancia de apilamiento dada por la ecuación (2).

4.2.1 Selección por torneo según operador de apilamiento (<c).

Este procedimiento reemplaza la selección usada en el algoritmo genético tradicional. Consiste en comparar dos soluciones las cuales poseen, cada una, dos atributos:

- Un rango de no dominación r_i. Según el frente de pareto.
- Una distancia local de apilamiento d_i.

La selección retorna la solución ganadora *i* basándose en dos criterios fundamentales.

- Si tiene mejor rango: $r_i < r_j$
- Si tienen el mismo rango pero *i* tiene mejor distancia de apilamiento: $d_i > d_j$

4.3 Determinación del Conjunto Descendiente Final

Antes de finalizar una generación del algoritmo se ejecuta un proceso de preselección y preservación de las soluciones de élite, que consiste en reunir el conjunto de soluciones Padres y los descendientes obtenidos por medio de los operadores de selección, cruce y mutación. De esta manera la población actual aumenta al doble de los individuos de la población inicial. Para ello es necesario clasificar el conjunto completo en sus respectivos frentes de dominancia y preservar los individuos que pertenezcan a los frentes de mejor calidad, tal como se ilustra en La Figura 5.

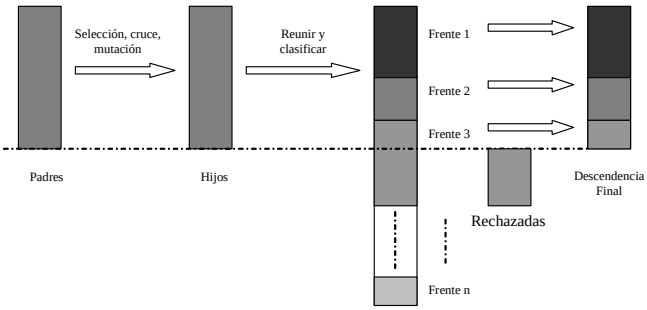


Figura 5. Determinación de la nueva Población

Si no es posible ingresar todas las alternativas de un frente determinado, entonces son eliminados aquellos individuos con una menor distancia de apilamiento dada por la ecuación (2).

4.4 Seudo-Código para el NSGA-II

1. Generar una población P de tamaño N.
2. Identificar los frentes de dominancia y evaluar las distancias de apilamiento en cada frente.
3. Usando selección (<c), cruzamiento y mutación se genera una población descendiente del mismo tamaño de P.
4. Reunir Padres e hijos en un conjunto de tamaño 2N y clasificar los frentes de dominancia.
5. Determinar el conjunto descendiente final seleccionando los frentes de mejor rango. Si se supera el límite de población N, eliminar las soluciones con menor distancia de apilamiento en el último frente seleccionado.
6. Sí se cumple el criterio de convergencia, Fin del proceso. De lo contrario retornar al paso 3.

5. CASOS DE PRUEBA

5.1 Problema Sin Restricciones

$$\begin{aligned} &\max f1 = 1.1 - x_1 \\ &\max f2 = 60 - \frac{1 + x_2}{x_1} \\ &0.1 \le x1 \le 1 \\ &0 \le x2 \le 5 \end{aligned} \qquad (3)$$

Los algoritmos genéticos son típicamente utilizados cuando las variables son de tipo entero, por lo que para el presente problema se requiere codificar las variables de tipo continuo por medio de una cadena de elementos binarios.

$x_1 =$ | 1 | 0 | 1 | 0 | 1 |

Figura 6. Representación de una variable continúa en 5 bits.

Si la cadena de bits de la figura 6 representa la variable x_1, implica que el intervalo [0.1;1] se encuentra dividido en (2^5-1) sub-intervalos iguales, con una resolución de 0,03226. A mayor cantidad de bits se obtiene una respuesta de mejor exactitud, sin embargo se aumenta considerablemente el espacio de soluciones.

Una alternativa de solución será formada entonces por un cromosoma que reúne cada una de las variables codificadas.

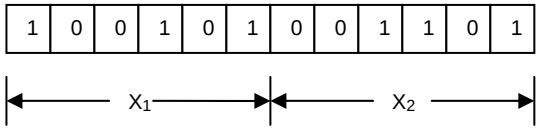


Figura 7. Representación de una alternativa de solución en 6 bits.

Si el bit más significante de cada variable se encuentra hacia el extremo izquierdo, significa que X_1=0,637 y X_2=1,048. La población de soluciones en cada generación se encontrará formada por configuraciones similares a la mostrada en la figura 7.

La Figura 8 resume los resultados obtenidos después de 50 generaciones del algoritmo, probabilidad de recombinación del 95%, tasa de mutación de 1% y tamaño de población de 50 individuos.

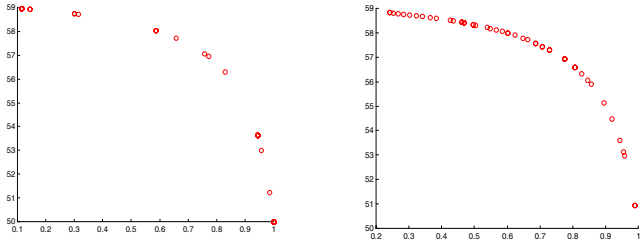


Figura 8. Frente de pareto resultante sin usar criterios de diversidad (izquierda), y usando el operador de distancia de apilamiento (derecha).

En ausencia del operador de diversidad la población tiende a homogenizarse alrededor de unos pocos individuos, lo cual muestra la necesidad de incorporar un operador adicional de diversidad dentro del algoritmo.

5.2 Problema con Restricciones

$$\begin{aligned}
&\min\ f_1 = x \\
&\min\ f_2 = y \\
&0 \geq -x^2 - y^2 + 0.1 \cdot Cos\left(16 \cdot Tan^{-1}\left(\frac{x}{y}\right)\right) \\
&\frac{1}{2} \geq \left(x - \tfrac{1}{2}\right)^2 + \left(y - \tfrac{1}{2}\right)^2 \\
&0 \leq x \leq \pi \\
&0 \leq y \leq \pi
\end{aligned} \qquad \textbf{(4)}$$

Una forma simple de resolver este problema es incorporando las restricciones dentro de las funciones objetivos por medio de una relajación de Lagrange. De esta manera un factor α asigna un valor que penaliza las infactibilidades y empeora la calidad de la función objetivo.

$$\begin{aligned}
&\min\ f_1 = x + \alpha \cdot factibilidad_1 + \beta \cdot factibilidad_2 \\
&\min\ f_2 = y + \alpha \cdot factibilidad_1 + \beta \cdot factibilidad_2 \\
&factibilidad_1 = -x^2 - y^2 + 0.1 \cdot Cos\left(16 \cdot Tan - 1\left(\frac{x}{y}\right)\right) \\
&factibilidad_2 = \left(x - \tfrac{1}{2}\right)^2 + \left(y - \tfrac{1}{2}\right)^2 - \frac{1}{2} \\
&0 \leq x \leq \pi \\
&0 \leq y \leq \pi \\
&\alpha = \begin{cases} 0 & si\ \ factibilidad_1 \leq 0 \\ 1 & de\ lo\ contrario \end{cases} \\
&\beta = \begin{cases} 0 & si\ \ factibilidad_2 \leq 0 \\ 1 & de\ lo\ contrario \end{cases}
\end{aligned}$$

La principal característica de este problema de optimización es que presenta un frente de pareto discontinúo, tal como se aprecia en la Figura 7, por lo cual asume un mayor grado de complejidad. La gráfica mostrada se logró a través de 100 generaciones de 200 individuos y una resolución de 5 bits por cada variable, y se puede apreciar en ella el espacio factible dado por la zona punteada, y la ubicación de las soluciones dominantes por medio de los círculos.

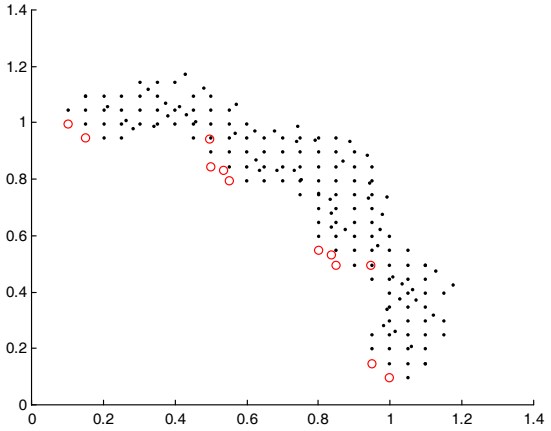


Figura 9. Espacio Factible del problema de optimización (puntos) con su respectivo frente de Pareto (círculos)

El proceso de optimización adquiere un mejor desempeño si se realiza una relajación del espacio de solución durante las primeras iteraciones, lo cual implica que las soluciones infactibles son consideradas como "*aceptables*" y participan en forma regular desde el punto de vista de los operadores genéticos. Posteriormente la región factible se aproxima hacia sus límites reales, de modo que los esquemas sobrevivientes contienen la

información de soluciones factibles y cercanas al frente dominante. Esto se logra asignando un valor de penalización relativamente bajo para las primeras iteraciones y aumentar sistemáticamente a medida que continúa el proceso, tal como se ilustra en la figura 10.

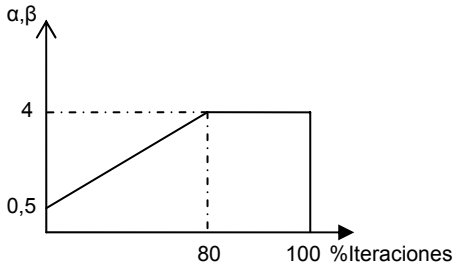
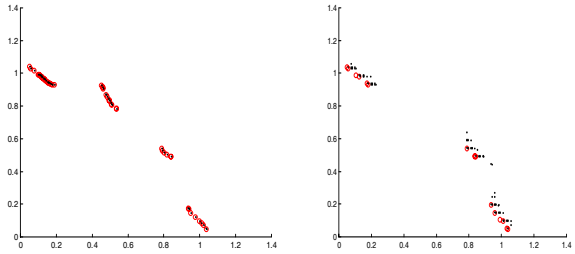


Figura 10. Valor de penalización a través del proceso de optimización.

Aplicando este esquema de penalización se logra un mayor número de soluciones pertenecientes al frente dominante y se genera el efecto elitista y envolvente que requiere el algoritmo multiobjetivo. Posteriormente el efecto de la penalización aleja las soluciones infactibles hacia frentes dominados muy alejados, por lo que se reduce progresivamente la posibilidad de que su información genética sobreviva en futuras generaciones.

Como caso extremo se implementó un esquema que asigna una penalización continua relativamente alta desde la primera generación del algoritmo, es decir eliminando de raíz soluciones infactibles. Sin embargo se observó un aumento considerable de las iteraciones necesarias para lograr el Frente de Pareto.



(a) (b)

Figura 11. Comparación de los Frentes de Pareto usando diferentes tipos de penalización.

En cada caso se asignaron los siguientes parámetros de búsqueda: tasa de mutación de 3%, población de 100 individuos, tasa de recombinación de 95% y 10 bits de resolución para cada variable. En el primer caso se eliminaron por medio de una fuerte penalización todas las combinaciones infactibles que resultaran, dando origen en 3000 generaciones al Frente mostrado en la Figura 11.b. En segunda instancia se implementó un esquema de penalización similar al de la Figura 10, logrando el Frente mostrado en la Figura 11.a en tan sólo 500 generaciones. Se puede apreciar que en este caso se produce un Frente de mejor calidad y con menor esfuerzo computacional al mostrado en la figura 11.b.

6. CONCLUSIONES Y RECOMENDACIONES

Es conveniente conocer ampliamente el tipo de problema que se quiere optimizar, y evaluar detenidamente si se trata de un caso multiobjetivo o sí es posible resolverlo adecuadamente por una técnica mono-objetivo. En caso contrario se tiene en el algoritmo elitista NSGA-II una herramienta de fácil asimilación si se tiene plena comprensión del concepto de dominancia y una buena base de la forma en que operan los algoritmos genéticos.

El algoritmo NSGA-II presenta la característica de mantener las soluciones pertenecientes al frente dominante dentro de los individuos de la población durante todas las generaciones del algoritmo, por lo que no se requiere que sean almacenadas en un archivo adjunto. Adicionalmente los mecanismos de diversidad son de rápida implementación, con exigencias computacionales relativamente bajas en comparación con otros algoritmos de su tipo, como el SPEA y DBGA.

La presencia de restricciones en el espacio de solución requiere de una manipulación adecuada de las infactibilidades. En ocasiones es conveniente realizar una relajación y aceptar la infalibilidad momentáneamente.

El esquema de penalización implementado logró un mayor número de soluciones pertenecientes al frente dominante.

La selección por torneo usando el operador de apilamiento, muestra mayor eficiencia que la selección por torneo tradicional en el campo multiobjetivo debido a que hace que la población evolucione de forma diversa y prefiriendo los individuos ubicados en los mejores frentes.

7. BIBLIOGRAFÍA

[1] Kalyanmoy Deb. “Multi-Objective Optimization using Evolutionary Algoritms”. Departamental of mechanical Enginnering. Indian Institute of technology, *Kanpur*, *India*. 2004.

[2] Coello Coello, Carlos, A. Van Veldhuizen, David A, Lamont Gary, B "Evolutionary Algorithms for Solving Multi-Objective Problems”. Volume 5 of the Book Series. Kluwer Academic Publishers, ISBN: 0306467623. Mayo 2002.

[3] Baran, Benjamn, et al.”Multi-Objective Optimization in Reactive Power Compensation”. Centro Nacional de Computación. Universidad Nacional de Asunción. San Lorenzo. Paraguay. 2006.

[4] Tanaka, M., Watanabe, H., Furuwaka, Y and Tanino, T. “GA-Based Decision Support System for Multicriteria Optimization. In Proceedings of the 1995 IEEE International Conference on Systems, Man, and Cybernetics, volume 2, Piscataway, NJ. IEEE.