UJAT
Universidad Juárez Autónoma de Tabasco
"Estudio en la duda. Acción en la fe"

DAIS 11111000011
División Académica de Informática y Sistemas

Doctorado Interinstitucional en Ciencias de la Computación

Tesis Doctoral

Optimización global con restricciones usando el algoritmo basado en el forrajeo de bacterias

Que presenta:
MCA. Betania Hernández Ocaña

Para obtener el grado de:

Doctora en Ciencias de la Computación

Comité tutoral:

Dra. Ma. Del Pilar Pozos Parra
Dr. Efrén Mezura Montes

Inteligencia Artificial en la Educación

Institución sede:
Universidad Juárez Autónoma de Tabasco

Cunduacán, Tabasco, México ***Octubre 2015***

Universidad Autónoma de Aguascalientes

Universidad Veracruzana

UJAT
Universidad Juárez Autónoma de Tabasco
"Estudio en la duda. Acción en la fe"

DAIS 11111000011
División Académica de Informática y Sistemas

Doctorado Interinstitucional en Ciencias de la Computación

Tesis Doctoral

Optimización global con restricciones usando el algoritmo basado en el forrajeo de bacterias

Que presenta:
MCA. Betania Hernández Ocaña

Para obtener el grado de:
Doctora en Ciencias de la Computación

Comité tutoral:

Dra. Ma. Del Pilar Pozos Parra
Dr. Efrén Mezura Montes

Jurado:

Dr. Francisco Javier Álvarez Rodríguez
Dr. José Adán Hernández Nolasco
Dr. Efrén Mezura Montes
Dra. Ma. Del Pilar Pozos Parra
Dr. Aleiandro Padilla Díaz

Inteligencia Artificial en la Educación

Institución sede:
Universidad Juárez Autónoma de Tabasco

Cunduacán, Tabasco, México ***Octubre 2015***

Universidad Autónoma de Aguascalientes

Universidad Veracruzana

Resumen

En este trabajo se propone modificar el Algoritmo de Optimización de Forrajeo de Bacterias (BFOA) para resolver problemas de optimización númerica con restricciones. BFOA pertenece al grupo de algoritmos que simulan el comportamiento de ciertas especies como aves, hormigas, abejas, peces, bacterias, conocido como algoritmos de inteligencia colectiva que recientemente han sido utilizados con mucho éxito en la solución de problemas de optimización numérica.

BFOA ha sido estudiado y aplicado a problemas complejos del mundo real como diseño de sistemas en Mecatrónica, segmentación de imágenes de resonancia magnéticas del cerebro, por mencionar algunos. Dentro de las conclusiones obtenidas en diversos trabajos por diferentes autores se comprueba que uno de los parámetros más sensibles del algoritmo es el tamaño de paso, que determina qué tanto avanza una bacteria cuando se desplaza en la búsqueda de nutrientes para alimentarse, el cual es calibrado por el usuario del algoritmo. Sin embargo, BFOA no está exento de caer en óptimos locales en cierto tipo de problemas de optimización y más aún cuando presentan restricciones, con lo cual la calidad del resultado y la convergencia del mismo se ven afectados. Dicho problema es común en los algoritmos de Computación Evolutiva e Inteligencia Colectiva.

Considerando estos antecedentes, las modificaciones a BFOA en este trabajo son: la autoadaptación del parámetro del tamaño de paso y la incorporación de un mecanismo de búsqueda local, con la finalidad de explorar y explotar la región de búsqueda tratando de evitar caer en óptimos locales y hacer una búsqueda adecuada que permita encontrar el óptimo global factible con una rápida convergencia. Además, la propuesta considera el uso de mecanismos para el manejo de restricciones ya que los problemas del mundo real frecuentemente presentan esta característica. Un mecanismo de sesgo también es considerado para distribuir a la población inicial hacia los límites y zonas centrales del espacio de búsqueda con el objetivo de hacer una búsqueda más rápida. Por último, un operador de mutación, usado en los algoritmos evolutivos, es adoptado en esta propuesta para mejorar la exploración y explotación el cual es implementado como operador de nado en el proceso quimiotáxico.

El nuevo algoritmo es sometido a un conjunto de experimentos, primero en problemas de prueba encontrados en la literatura especializada y posteriormente otros particulares de optimización global con restricciones como sistemas de diseños Mecatrónico y un generador de menús nutricionales. Los resultados son analizados y comparados contra algoritmos del estado del arte usando medidas de desempeño encontradas en la literatura especializada. Finalmente, un análisis desde el punto de vista mecánico es integrado cuando la propuesta es probada en tres problemas de diseño Mecatrónico derivados de la Síntesis de un mecanismo plano de cuatro barras. En cada uno de los experimentos la propuesta obtuvo un rendimiento competitivo frente a su versión original y a otros algoritmos del estado del arte

Abstract

This work proposes to amend the Bacteria Foraging Optimization Algorithm to solve constrained numerical optimization problems. BFOA belongs to the group of algorithms that simulate the behavior of certain species such as birds, ants, bees, fish, bacteria, these known as collective intelligence algorithms that have recently been used with great success in solving numerical optimization problems.

BFOA has been studied and applied to complex real-world problems as Mechatronic systems design, segmentation of magnetic resonance images of the brain, to name a few. In the conclusions of several studies by different authors it is found that one of the most sensitive user-defined parameters of the algorithm is the step size, which determines the movement distance of a bacterium when swims in search of nutrients to feed. However, BFOA presents premature convergence in certain types of optimization problems and even more when the problems presenting constraints, whereby the quality of the result and the convergence of it are affected. Premature convergence is a problem common in algorithms Evolutionary Computation and Collective Intelligence.

Considering this background, the modifications to BFOA in this work are: self-adaption of the step-size parameter, add a local search operator in order to explore and exploit the search region trying to avoid falling into optimum local and find the global optimum feasible with rapid convergence. Moreover, the proposal also considers a mechanisms for handling constraints because the real-world problems often have this feature. A skew mechanism is also considered to distribute the initial population to the limits and central areas of the search space with of aim of improve of performance of algorithm from the first generations. Finally, a mutation operator, used in evolutionary algorithms, is incorporated as a swim operator in the chemotaxis process.

The new proposal is subjected to a set of experiments, first test problems found in the literature and subsequently other particular constrained numerical optimization problems as Mechatronic systems designs and a nutritional menus generator. The results are analyzed and compared with state-of-the- art algorithms using performance measures found in the literature. Finally, an analysis from a mechanical point of view is integrated when the proposal was tested in three Mechatronic design problems arising from the synthesis of a flat four-bar mechanism. In each of the experiments, the proposal obtained a competitive performance when was compared against its original version and other state-of-the-art algorithms.

Agradecimientos

En construcción...

Publicaciones

En el siguiente listado se mencionan cada uno de los artículos derivados de la investigación.

1. Betania Hernández-Ocaña, Efrén Mezura-Montes and Ma. Del Pilar Pozos-Parra, A review of the bacterial foraging algorithm in constrained numerical optimization, in Proceedings of the 2013 IEEE Congress on Evolutionary Computation, Cancún, México, pages: 2695–2702, IEEE Press, June, 2013, ISBN:978-1-4799-0451-8.

2. Betania Hernández-Ocaña, Pilar Pozos-Parra, and Efrén Mezura-Montes. Stepsize control on the modified bacterial foraging algorithm for constrained numerical optimization. In Proceedings of the 2014 Conference on Genetic and Evolutionary Computation, GECCO '14, pages 25-32, New York, NY, USA, 2014. ACM.

3. Betania Hernández-Ocaña, María-del-Pilar Pozos-Parra and Efrén Mezura-Montes, Improved Modified Bacterial Foraging Optimization Algorithm to solve constrained numerical optimization problems, Applied Mathematics and Information Sciences, (in press), 2015.

4. Betania Hernández-Ocaña, María-del-Pilar Pozos-Parra, Efrén Mezura-Montes, Edgar Alfredo Portilla-Flores, Eduardo Vega-Alvarado y Maria Bárbara Calva-Yáñez. Two swim operators in the Modified Bacterial Foraging Algorithm for the Optimal Synthesis of Four-Bar Mechanisms, Computer Methods in Applied Mechanics and Engineering, (In review), 2015.

Contenido

Lista de figuras

Lista de tablas

Índice de algoritmos

Capítulo 1

Introducción

En este capítulo se describen los antecedentes encontrados en la literatura especializada en referencia al tema de investigación de este trabajo. Además se da a conocer la motivación que dio origen a esta investigación, los alcances de la investigación, las contribuciones y finalmente, la organización de la tesis.

1.1. Antecedentes

En la actualidad los algoritmos meta-heurísticos son una opción popular para resolver problemas complejos de optimización [1]. Entre ellos se tienen a los algoritmos evolutivos (AE), cuyo funcionamiento se basa en emular el proceso de evolución natural y la supervivencia del más apto, para que de esta manera se puedan muestrear espacios de búsqueda complejos [2]. Por otro lado, a mediados de los 1990's surge un nuevo grupo de algoritmos meta-heurísticos inspirados también en fenómenos encontrados en la naturaleza. A este conjunto de algoritmos se les agrupa en el área llamada Inteligencia Colectiva (IC) y su característica general es que basan su funcionamiento en comportamientos sociales y cooperativos de organismos simples como insectos y aves [3].

El algoritmo de IC más popular para resolver Problemas de Optimización Numérica con Restricciones (PONR), es el llamado optimización mediante cúmulos de partículas (PSO por sus siglas en inglés, Particle Swarm Optimization) [4]. PSO ha sido ampliamente estudiado y aplicado en la resolución de problemas de optimización con restricciones [5, 6, 7, 8, 9]. Sin embargo, existen algoritmos de IC más recientes que aún no han sido estudiados con profundidad como el algoritmo basado en el forrajeo de bacterias (BFOA por sus siglas en inglés, Bacterial Foraging Optimization Algorithm). A partir de las ideas iniciales de Bremermann [10], en el año 2002 Passino propone BFOA [11] en el cual cada bacteria trata de maximizar su energía obtenida por unidad de tiempo empleada en el proceso de forrajeo, donde también evade sustancias nocivas. Más aún, las bacterias se pueden comunicar entre sí mediante la segregación de sustancias. Un cúmulo de bacterias S se comporta de la siguiente manera [11].

1. Las bacterias se distribuyen de manera aleatoria en el mapa de nutrientes.
2. Las bacterias se mueven hacia regiones con altos contenidos de nutrientes en el mapa mediante un proceso de movimiento basado en elementos químicos (quimiotáxis) que contempla los movimientos de giro y nado de cada bacteria. Las bacterias que se localicen en regiones con sustancias nocivas morirán y las ubicadas en zonas con baja concentración de nutrientes se dispersarán. Finalmente, aquellas localizadas en regiones con altos contenidos de nutrientes se reproducirán.

3. Las bacterias localizadas en las regiones con altos contenidos de nutrientes tratarán de atraer a otras bacterias mediante la segregación de atractores químicos.

4. El cúmulo de bacterias se ubica en la región con más altos contenidos de nutrientes.

5. Las bacterias se dispersan para buscar más nutrientes en el mapa.

En BFOA se tienen cuatro procesos principales: (1) quimiotáxis (nado-giro), (2) agrupamiento, (3) reproducción y (4) eliminación-dispersión. Las bacterias son soluciones potenciales al problema y su ubicación representa los valores de las variables de decisión del problema. Las bacterias pueden moverse (generar nuevas soluciones) mediante el ciclo quimiotáxico; se genera además un movimiento mediante la atracción que ejercen soluciones en zonas prometedoras del espacio de búsqueda, se permite la reproducción de las mejores soluciones y finalmente se eliminan del cúmulo aquellas bacterias localizadas en zonas de baja calidad.

El algoritmo BFOA ha sido aplicado en diversos problemas como: optimización global sin restricciones [12, 13, 14, 15], optimización multimodal [16, 17] predicción [18], optimización dinámica [19], aprendizaje computacional [20], sistemas dinámicos [21], sistemas de potencia [22, 23], procesamiento de señales [24], agrupamiento de datos [25], video compresión [26], entre otros.

Por otro lado, se tienen reportados estudios sobre los componentes de BFOA, como lo son: el proceso de reproducción [27] y el tamaño de paso en el ciclo quimiotáxico [28, 29], e incluso existen reportes de la combinación de BFOA con otros algoritmos meta-heurísticos como un Algoritmo Genético [30], PSO [31, 32] y Evolución Diferencial [33] con el fin de mejorar su rendimiento. Sin embargo, la convergencia prematura es una desventaja de los métodos de optimización basado en cúmulos. BFOA ha sido combinado con otros algoritmos de búsqueda como el método de búsqueda $Hooke - JeevesPattern$ como operador de búsqueda local [34]. Por otra parte, BFOA se combinó con las reglas *Armijo* como buscador local en la propuesta denominada algoritmo de Optimización Espiral de Forrajeo de Bacterias (sus siglas en Ingles SBFO) el cual es un multi-agente, basado en el algoritmo de gradiente que minimiza tanto la función objetivo principal (costo local) y la distancia entre cada agente y un punto central temporal (costo global). Valores aleatorios en los parámetros se adoptaron en esta propuesta para hacer frente a la convergencia prematura [35].

En lo referente a propuestas de solución de instancias del problema de optimización global con restricciones, en 2009 se propuso una versión simplificada de BFOA llamada Modified BFOA (MBFOA) en [36]. En MBFOA se tiene un mecanismo para el manejo de las restricciones basado en reglas de factibilidad [37] y una disminución de parámetros respecto a los del BFOA original. Además MBFOA se aplicó a un conjunto de problemas de diseño en ingeniería química e ingeniería mecánica obteniendo resultados competitivos. Por otro lado, MBFOA se aplicó a la resolución de un problema de diseño mecánico bi-objetivo en presencia de restricciones en [38].

El problema de optimización global con restricciones es uno de los problemas para el que no se reporta mucho trabajo en su resolución usando BFOA [39], según la revisión del estado del arte que se detalla en el Capítulo 2. Este tipo de problema también conocido como el problema general de programación no-lineal se puede definir como:

$$\begin{gathered}\text{Minimizar } f(\vec{x}) \text{ sujeta a:} \\ g_i(\vec{x}) \leq 0,\ i = 1, ..., m, \\ h_j(\vec{x}) = 0,\ j = 1, ..., p,\end{gathered}$$

Donde $\vec{x} \in R^n$ tal que $n \geq 1$, es el vector de soluciones $\vec{x} = [x_1, x_2, ..., x_n]^T$, donde cada $x_i, i = 1, ..., n$ está delimitada por el límite inferior y superior $L_i \leq x_i \leq U_i$; m es el número de restricciones de desigualdad y p es el número de restricciones de igualdad (en ambos casos, las restricciones podrían ser lineales o no lineales). Si denotamos con F a la región factible (donde se encuentran todas las soluciones que satisfacen al problema) y con S a todo el espacio de búsqueda, entonces debe ser claro que $F \subseteq S$.

Tanto los algoritmos evolutivos como los de IC carecen, en sus versiones originales, de un mecanismo para manejar las restricciones del problema. En otras palabras, son optimizadores para espacios no restringidos. De ahí que el diseño de un mecanismo para manejar la factibilidad o no de una solución es un problema abierto en el área. En [40] se encuentra una revisión exhaustiva y actualizada del estado del arte en optimización con restricciones usando AEs y ICs.

La técnica más popular para manejar restricciones en un algoritmo EA o un IC es la función de penalización [41], donde se castiga la aptitud de las soluciones no factibles para dar preferencia a aquellas que si cumplen con las restricciones del problema. La principal desventaja de las funciones de penalización es que el usuario debe definir los llamados factores de penalización, cuyos valores determinan la severidad del castigo a las soluciones no factibles; pues se ha reportado que éstos son dependientes de las características o complejidad del problema [42].

Por otro lado, en los últimos años ha tomado popularidad un conjunto de reglas simples basadas en factibilidad que no requieren parámetros adicionales que el usuario deba definir y cuya adición a los algoritmos es sencilla [37].

1.2. Motivación

De la revisión a la literatura especializada, que se detalla en el Capítulo 2 de este documento y de los antecedentes, se observa lo siguiente:

- Aunque se reportan aplicaciones diversas, no se reporta una versión basada en BFOA para resolver específicamente problemas de optimización global con restricciones. El caso de excepción es MBFOA que resuelve el problema mono-objetivo con restricciones, pero no ha sido probado extensivamente.
- Sólo se ha probado reglas de factibilidad como técnica de manejo de restricciones en MBFOA.
- No se reportan estudios de sensibilidad de parámetros en MBFOA.
- No se tienen reportes de estudios avanzados en optimización global con restricciones basados en BFOA.

Derivado de estas observaciones surgen un conjunto de preguntas las cuales deben responderse en base a estudios y análisis del algoritmo BFOA:

1. ¿Se reportan en la literatura especializada estudios amplios y profundos sobre el algoritmo de optimización basado en el forrajeo de bacterias para resolver problemas de optimización global con restricciones?
2. ¿Qué mecanismo del algoritmo basado en el forrajeo de bacterias, susceptible de modificación, mejora el rendimiento del algoritmo?

3. ¿Qué buscador local existente en la literatura especializada usado en problemas de optimización con restricciones mejorará el rendimiento del algoritmo del forrajeo de bacterias?

1.3. Alcances de la investigación

La hipótesis a comprobar en este trabajo se define como:

Mediante la incorporación de un mecanismo de búsqueda local y la adaptación/autoadaptación de parámetros al algoritmo basado en el forrajeo de bacterias, es posible resolver problemas de optimización global con restricciones y obtener resultados competitivos, comparado con los algoritmos del estado del arte, usando medidas de desempeño encontradas en la literatura especializada, donde se incremente en un 5 % el porcentaje de progreso dentro de la zona factible.

Con dicha hipótesis se debe hacer un primer estudio y modificación de mecanismos del BFOA que lo hagan más competitivo al resolver PONR y obtener el primer algoritmo híbrido (búsqueda global-local) basado en BFOA para resolver PONR, para lo cual se tiene como objetivo general:

- Implementar un algoritmo basado en el forrajeo de bacterias para resolver problemas de optimización global con restricciones, que sea competitivo con respecto a los algoritmos metaheurísticos del estado del arte y que abarque aspectos poco explorados y potencialmente competitivos dentro de esta área como búsqueda local y adaptación/autoadaptación de parámetros.

Y como objetivos específicos que permitan llegar al objetivo general y por consiguiente comprobar la hipótesis planteada son:

- Determinar qué combinación del algoritmo con una técnica de manejo de restricciones existente en la literatura mejora su desempeño.
- Modificar el (los) mecanismo(s) del algoritmo propuesto con la finalidad de mejorar su desempeño de manera sustancial al resolver problemas de optimización con restricciones.
- Determinar la combinación del algoritmo propuesto con un buscador local existente en la literatura utilizado en problemas de optimización, con la finalidad de mejorar el desempeño del algoritmo.
- Implementar el algoritmo resultante para resolver problemas de optimización con restricciones, entre los que se contemplan diseños del área de Mecatrónica y la generación de menús nutricionales para un módulo de dominio de un sistema tutor inteligente.

1.4. Contribuciones

- El primer survey sobre BFOA en optimización numérica
- El primer estudio y modificación de mecanismos del BFOA que lo hacen más competitivo al resolver el problema de optimización con restricciones
- El primer algoritmo híbrido (búsqueda global-local) basado en BFOA para resolver el problema de optimización con restricciones
- El primer algoritmo basado en BFOA con mutación como operador de nado para resolver el problema de optimización con restricciones

1.5. Organización de la tesis

El documento de tesis ésta organizado como sigue:

- En el Capítulo 1 se describen los antecedentes encontrados en la literatura especializada en referencia al tema de investigación de este trabajo. Además se da a conocer la motivación que dio origen a esta investigación, los alcances de la investigación, las contribuciones y finalmente, la organización de la tesis es presentada.
- En el Capítulo 2 se mencionan brevemente las propuestas basadas en BFOA para resolver problemas de optimización con y sin restricciones encontradas en la literatura especializada, finalmente se provee un análisis de las propuestas revisadas.
- En el Capítulo 3 se describe el algoritmo modificado basado en el forrajeo de bacterias.
- En el Capítulo 4 se mencionan las técnicas de manejo de restricciones y se reportan las adaptaciones realizadas al algoritmo MBFOA para conocer qué técnica de manejo de restricciones permite mejorar su rendimiento en el proceso de optimización.
- En el Capítulo 5 se describen cuatro mecanismos para el calculo del tamaño de paso, los cuales son: dinámico, adaptativo, estático y dinámico.
- En el Capítulo 6 se describen cada una de las adaptaciones realizadas al algoritmo modificado basado en el forrajeo de bacterias para mejorar su rendimiento en espacios restringidos en la propuesta llamada IMBFOA.
- En el Capítulo 7 se describe a la propuesta llamada TS-MBFOA (por sus siglas en Inglés Two Swim-Modified Bacterial Foraging Optimization Algorithm) y las implementaciones realizadas en problemas de prueba y del mundo real.
- En el Capítulo 8 se presentan las conclusiones y los trabajos futuros a realizar.

Capítulo 2

Optimización global basada en forrajeo de bacterias

BFOA es un algoritmo que ha sido utilizado para resolver problemas de optimización numérica, con y sin restricciones, tanto en funciones de prueba como en problemas particulares modelados como un problema de programación no-lineal (descrito en la sección 1.1). En los siguientes apartados se mencionan las técnicas de manejo de restricciones y las propuestas basadas en BFOA para resolver problemas de optimización con y sin restricciones encontradas en la literatura especializada, finalmente se provee un análisis de las propuestas revisadas.

2.1. Implementación de BFOA en problemas de optimización numérica

Esta sección es un resumen de la revisión del estado del arte referente a las diversas implementaciones de BFOA para resolver problemas de optimización numérica con y sin restricciones encontrada en la literatura especializada. Además se incluye un análisis para dar a conocer lo más sobresaliente de esta revisión del estado del arte.

2.1.1. Optimización global sin restricciones basada en BFOA

Chatterjee y Matsuno [43] implementan BFOA para resolver un problema mono-objetivo sin restricciones que consiste en optimizar los parámetros de un sistema de lógica difusa para la localización simultánea y mapeo (SLAM) de robot móviles y vehículos autónomos. Los resultados obtenidos por BFOA fueron comparados con el sistema EKF donde BFOA mejora el rendimiento de la localización, es decir encuentra mejor combinación de parámetros para el sistema difuso. Sin embargo no se menciona el número de ejecuciones independientes realizadas para obtener dichos resultados.

Kim y Cho [44] proponen un hibrido basado en BFOA, lógica difusa y sistema inmune de selección clonal para resolver un problema mono-objetivo sin restricciones. La propuesta es realizar en el proceso quimiotáxico la selección de los mejores individuos de la población con base en el valor de aptitud. Estos individuos son clonados y se les genera mutaciones, los clones mutados son ordenados de forma decreciente con base en su valor de aptitud. Los individuos y los clones mutantes son comparados quedando aquellos que son mejores. La propuesta es probada para optimizar el vector del sistema de control de un motor de inducción. Los resultados de la propuesta son competitivos al compararlos con BFOA y un AG. Los autores no mencionan el número de ejecuciones independientes realizadas, tampoco detallan cómo incluyen la lógica difusa en el hibrido ya que en la conclusión

solo mencionan que fue utilizada para extender el tamaño de paso frente a condiciones de cambio, haciendo complicada la interpretación de su propuesta. Se agregan cinco parámetros más a los propios de BFOA que el usuario tiene que calibrar, los cuales son propios del algoritmo sistema inmune de selección clonal.

Tang et al. [13] modifican BFOA calculando el tamaño de paso en el nado y giro de la bacteria para resolver conjunto de funciones de pruebas de optimización mono-objetivo sin restricciones. La propuesta consiste en calcular el tamaño de paso de dos formas, 1) cuando la bacteria gira, usando un tamaño de paso inicial y el vector de longitud de los límites de dominio de la búsqueda. 2) Cuando la bacteria nada, se hace uso del vector de longitud de los límites de dominio de la búsqueda, una constante definida por el usuario y un número aleatorio. Después del ciclo quimiotáxico las bacterias son ordenadas ascendentemente de acuerdo a la suma de aptitud durante el tiempo de vida de la bacteria, donde la bacteria con mejor suma de aptitud es conservada y el resto de la población es seleccionada aleatoriamente. La propuesta fue probada en siete funciones de prueba y los resultados obtenidos de 20 ejecuciones independientes fueron mejor en cuatro de las funciones de prueba al compararlos con otros algoritmos encontrados en la literatura. La propuesta requiere de la calibración de cuatro parámetros más a los propios de BFOA, el proceso de reproducción y eliminación-dispersión son eliminados y se realiza el proceso de atracción hacia la mejor bacteria de la población.

Kim et al. [30] proponen un hibrido de BFOA y un AG para resolver un problema de optimización mono-objetivo sin restricciones. En esta propuesta se lleva a cabo mutación dinámica y cruza en BFOA. La propuesta fue probada en cuatro funciones de prueba para determinar qué tamaño de paso provee un mejor rendimiento al algoritmo propuesto, asimismo se comparan los resultados con un AG, donde la propuesta obtiene mejores resultados y una rápida convergencia. Posteriormente, la propuesta es probada en un problema para afinar un controlador regulador de voltaje automático AVR. Los resultados obtenidos son comparados con los algoritmos PSO, AG, AG-PSO, donde la propuesta es competitiva con la solución encontrada para la función objetivo, sin embargo, los algoritmos con los que fue comparado encuentran soluciones mejores. No se especifican el número de ejecuciones independientes realizadas al algoritmo, también no se detalla en qué momento y cómo se lleva a cabo la mutación dinámica y la cruza en BFOA. Al no presentar los parámetros de número de ciclos de reproducción, número de eventos de eliminación-dispersión y la probabilidad de eliminación-dispersión, se puede pensar que estos procesos no se llevan a cabo en la propuesta. Los parámetros de probabilidad de mutación y cruza en la propuesta son nuevos parámetros que deben ser calibrados por el usuario.

Biswas et al. [31] proponen un hibrido de BFOA y PSO para resolver un conjunto de funciones de prueba de optimización mono-objetivo sin restricciones. Dentro del BFOA al final del ciclo quimiotáxico, todas la bacterias son dirigidas a la posición de la mejor bacteria de la población del ciclo en curso con el operador de generación de dirección de PSO. Seguido se llevan a cabo los procesos de reproducción y eliminación-dispersión. El hibrido fue probado en cinco funciones de prueba, los resultados obtenidos de 25 ejecuciones independientes con diferente dimensionalidad fueron comparados con algoritmos encontrados en la literatura incluidos BFOA y PSO, donde el hibrido obtiene mejor resultado para la función objetivo de cada problema probado, además de una rápida convergencia y menor desviación estándar de los resultados. La propuesta es interesante, sin embargo, la incorporación de más parámetros a calibrar (propios de PSO) por el usuario es una desventaja.

Estos mismos autores en [45] proponen un hibrido entre BFOA y ED para resolver un conjunto de funciones de prueba de optimización mono-objetivo sin restricciones. En éste hibrido sólo se lleva a

cabo el proceso quimiotáxico de BFOA, donde al movimiento de nado le es incluido un factor de inercia hacia la posición previa de la bacteria, y el tamaño de paso es calculado de acuerdo al valor de la función de aptitud de la bacteria y a variables constantes. En cada ciclo quimiotáxico cada bacteria es mutada y cruzada con bacterias de la población de acuerdo a ED. El hibrido fue probado en seis funciones de prueba y los resultados obtenidos de 30 ejecuciones independientes fueron comparados con los algoritmos BFOA, una propuesta derivada de ED llamada ED/rand/1/bin, un hibrido de BFOA y un AG, donde la propuesta obtuvo mejor resultado a la función objetivo de cada problema. Cabe mencionar que la propuesta es simple de implementar, los parámetros de BFOA se reducen al no implementar el proceso de reproducción y eliminación-dispersión, pero son agregados los parámetros propios de ED para la mutación y cruza.

Acharya et al. [46] implementan BFOA para resolver un problema de optimización mono-objetivo sin restricciones que consiste en minimizar la dependencia entre los componentes de salida de la técnica de procesamiento de señal estática, que tiene como objetivo encontrar una representación lineal de datos no Gaussianos. No se mencionan el número de ejecuciones independientes realizadas por BFOA pero los resultados mostrados en tablas y gráficas muestran que BFOA obtiene una mejor línea de los datos no Gaussianos en comparación a los otros dos algoritmos encontrados en la literatura, sin embargo, en convergencia BFOA es más lento que uno de los algoritmos con los que se compara.

Sumanbuba et al.[47] implementan BFOA para resolver un problema mono-objetivo sin restricciones que consiste en ajustar los parámetros de un estabilizador para un sistema de alimentación multi banda. Los resultados obtenidos por BFOA, al compararlos con una propuesta de afinación, son mejores de acuerdo a lo mostrado en las gráficas de estabilidad y rapidez de los generadores de energía. No se menciona el número de ejecuciones independientes realizadas al algoritmo ni los parámetros utilizados por BFOA.

Majhi y Panda [21] implementan BFOA para resolver un problema mono-objetivo sin restricciones que busca la identificación de un sistema no-lineal dinámico. BFOA es utilizado para actualizar los pesos del modelo FLANN que es una red neuronal que permite la identificación de un sistema no-lineal dinámico. Los resultados obtenidos son comparados con un modelo encontrado en la literatura, donde BFOA converge con menos iteraciones y el costo computacional es reducido, ambas propuestas obtienen resultados similares de acuerdo a lo presentado en las gráficas. Sin embargo el número de parámetros propios de BFOA definidos por el usuario son una desventaja.

Estos mismos autores y otros [18] implementaron BFOA para optimizar los pesos de conexión del modelo lineal adaptativo que es un problema mono-objetivo sin restricciones con el fin de hacer predicciones del mercado de valores S&P 500 y DJIA. No se menciona el número de ejecuciones independientes realizadas pero los resultados obtenidos con BFOA son comparados con un modelo encontrado en la literatura llamado MLP, donde BFOA es computacionalmente más eficiente al realizar menos iteraciones, resultados más exactos y se requiere de menos tiempo para la espera de resultados.

Hanmandlu et al. [48] implementan BFOA para resolver un problema mono-objetivo sin restricciones que consiste en identificar números Hindi escritos a mano usando el método de caja. Los autores no comparan sus resultados debido a que no existe una metodología para la comparación de experimentos de este tipo de problemas, sin embargo, los autores mencionan que es posible mediante BFOA identificar los números Hindi aunque en números como 2 y 3, 0 y 9 el algoritmo presenta confusión.

Bakwad et al. [17] modifican al BFOA adaptando el tamaño de paso para resolver problemas de optimización mono-objetivo sin restricciones. En esta propuesta el tamaño de paso es determinado por la diferencia del valor de la función de aptitud previa de la bacteria *p1* y la posición actual de la bacteria *p2*, si la diferencia es positiva entonces la bacteria avanzará en dirección de *p1*, de lo contrario realiza un giro aleatorio. En esta propuesta la evaluación de la función objetivo en cada bacteria puede ser secuencial o paralela. La propuesta fue probada en tres funciones de prueba, donde los resultados fueron comparados con BFOA y estos fueron mejores en obtener mejor solución a la función objetivo de los problemas, precisión y menor tiempo computacional.

Chu et, al. [14] modifican al BFOA adaptando el tamaño de paso y modificando la forma en que nadan las bacterias utilizando un factor de atracción como el utilizado en PSO para resolver problemas de optimización mono-objetivo sin restricciones. El tamaño de paso de esta propuesta es calculado con un tamaño de paso inicial y una constante de control de creciente en el proceso de reproducción y eliminación-dispersión. Cuando una bacteria decide nadar en el proceso quimiotáxico, esta es dirigida hacia la mejor bacteria de la población con un factor de atracción definido por el usuario, donde la mejor bacteria es aquella que tiene mejor valor en la función de aptitud. La propuesta fue probada en trece funciones de prueba y comparada con los algoritmos BFOA y PSO con los resultados de diez ejecuciones independientes. La propuesta obtiene mejor resultado con respecto a la desviación estándar en todos los problemas probados. Los autores mencionan que la propuesta no es eficiente para problemas de optimización de baja dimensionalidad y se tienen dos parámetro más definidos por el usuario, sin embargo es mejor en cuanto a rapidez de convergencia y precisión de resultados en comparación que BFOA y PSO.

Data et al. [49] adaptan el tamaño de paso del BFOA para resolver un problema mono-objetivo sin restricciones. La propuesta consiste en calcular el tamaño de paso que utiliza la bacteria en cada ciclo quimiotáxico tomando en cuenta el resultado de los tres últimos tamaños de paso, esta propuesta es basada en el principio de modulación delta que corresponde obtener la desviación entre la señal real y la señal modulada captada por las antenas. La propuesta es probada en el problema que busca optimizar la amplitud y el peso de las fases de una colección de antenas lineales para maximizar el número de colecciones en cualquier dirección deseada y nulos en direcciones específicas. No se reportan el número de ejecuciones independientes realizadas a la propuesta, tampoco se comparan sus resultados contra otras meta-heurísticas y sólo se presenta el resultado obtenido por la propuesta al problema, la cual es optimizada con éxito según los autores. Una de las características de esta propuesta es el manejo de memoria de las bacterias para registrar sus últimos tres tamaños de paso.

Maitra y Chatterjee [50] implementan BFOA para resolver un problema mono-objetivo sin restricciones que consiste en maximizar el criterio de entropía para el alumbramiento de segmentación de imágenes de resonancia magnéticas del cerebro basado en histograma. No se menciona el número de ejecuciones independientes realizadas a BFOA pero se muestran resultados de nueve imágenes tratadas con 3 y 5 alumbramientos, además se presentan los resultados de probar a BFOA con diferentes valores en los parámetros de tamaño de población, número de ciclos quimiotáxicos y número de nados permitidos, para determinar que combinación de parámetros obtiene mejores resultados para el problema. Los resultados de BFOA son mejores comparados con una propuesta basada en PSO al obtener valores mayores para la función objetivo del problema.

Korani [32] propone un hibrido entre BFOA y PSO para resolver un problema mono-objetivo sin restricciones. El hibrido consiste en orientar la dirección de nado de la bacteria con el operador de generación de dirección de PSO. La dirección de la bacteria se calcula en base a la información guardada de la mejor posición que ha logrado la bacteria, información de la posición actual, in-

formación de la posición de la mejor bacteria de la población (mejor valor de aptitud), coeficiente de aceleración, factor de inercia y el vector de velocidad. El hibrido fue probado en el problema de afinación del controlador proporcional integral y derivativo (PID), los resultados obtenidos fueron comparados con BFOA y PSO, donde el hibrido fue quien obtuvo un mejor valor a la función objetivo del problema además de una rápida convergencia. Este hibrido es simple de implementar pero son agregados los parámetros propios de PSO donde algunos de ellos (coeficiente de aceleración y factor de inercia) tienen que ser definidos por el usuario aparte de los parámetros propios de BFOA.

Dasgupta et al. [29] adaptan el tamaño de paso de BFOA para resolver una serie de problemas de optimización mono-objetivo sin restricciones. Los autores proponen dos formas de adaptar el tamaño de paso, 1) dividiendo 1 entre la suma de 1 más una constante positiva dividida entre el valor absoluto del valor de aptitud de la bacteria en proceso. 2) Usando la forma 1) pero el valor absoluto es la diferencia del valor de aptitud de la bacteria y el valor de aptitud de la mejor bacteria de la población. Ambas propuestas fueron probadas en diez funciones de prueba encontradas en la literatura, los resultados obtenidos de cincuenta ejecuciones independientes fueron comparados con cinco algoritmos encontrados en la literatura entre ellos BFOA, donde la propuesta 1 de este trabajo obtiene en la mayoría de los problemas mejor resultado, desviación estándar y convergencia.

Chen et al. [51] modifican al BFOA adaptando el tamaño de paso, además de implementar reinicialización de posición de las bacterias y dividir el espacio de búsqueda para mejorar su exploración y explotación. La propuesta es utilizada para resolver problemas de optimización mono-objetivo sin restricciones. De este trabajo se derivan dos propuestas, la primera (CBFO-S) realiza fases evolutivas e implementan BFOA. Al final de la fase, cada bacteria es reinicializada a la mejor posición que visitó durante el proceso quimiotáxico, cabe mencionar que la mejor posición depende del valor de aptitud. El tamaño de paso decrece en cada fase al ser dividido entre una variable constante definida por el usuario mayor a 1. La segunda propuesta (CBFO-H) se implementa BFOA, después cada bacteria es reinicializada a la posición de la mejor solución encontrada individualmente asociándole un tamaño de paso pequeño. El espacio de búsqueda del problema es dividido en 2 y por cada subespacio se realiza el BFOA, la mejor solución encontrada en cada subespacio actualiza la solución global del problema siempre y cuando sea mejor a la que se tiene almacenada. Por cada ciclo quimiotáxico el tamaño de paso decrece al ser dividido entre una variable constante definida por el usuario mayor a 1, en determinados ciclos, también definido por el usuario. Estas dos propuestas fueron probadas en cuatro funciones de prueba. Los resultados obtenidos de treinta ejecuciones independientes fueron comparados con el original BFOA, un PSO y un AG, donde CBFO-H obtiene mejores resultados para la función objetivo en tres problemas y sólo en uno es superado por CBFO-S. En estas propuestas el tamaño de paso no es constante, con esto se permite la exploración y explotación del espacio de búsqueda, sin embargo se tienen que inicializar el tamaño de paso superior y el tamaño de paso inferior, los cuales decrecen en relación a dos variables constantes definidas por el usuario que hacen más complejo al algoritmo al momento de ajustarlo, además del nuevo ciclo llamado fases evolutivas hace que el BFOA se realice n veces por cada ejecución y por cada división del espacio de búsqueda se realiza el BFOA lo cual genera que el costo computacional se incremente. Asimismo se requiere que las bacterias tengan memoria para registrar las direcciones en las que ha estado del espacio de búsqueda.

Dasgupta et al. [15] eliminan el proceso de reproducción para evitar la saturación (pérdida de diversidad) y convergencia prematura para resolver problemas de optimización mono-objetivo sin restricciones con alta dimensionalidad. Una de las características de esta propuesta es su pequeña población (solo tres bacterias). Después de un ciclo completo del proceso quimiotáxico las tres bacterias son ordenadas de acuerdo al valor de aptitud. La mejor bacteria (*rank* 1) conserva su posición, la segunda mejor bacteria (*rank* 2) se mueve a una posición muy cercana de la bacteria con *rank* 1 con

el propósito de facilitar la búsqueda local y la peor bacteria (*rank* 3) es inicializada aleatoriamente en el espacio de búsqueda para mantener la diversidad de la población y evitar la convergencia prematura. Se realizaron cincuenta ejecuciones independientes para probar la propuesta en cinco funciones de prueba. Los resultados son comparados con el BFOA original, donde los resultados de la propuesta son mejores. La principal ventaja de esta propuesta es su bajo costo computacional y tiempo de cálculo. Sin embargo no se compara contra otras meta-heurísticas y no se conoce si los resultados son competitivos y de calidad ante otras propuestas.

Vaisakh et al. [52] proponen un hibrido de PSO y BFOA para resolver un problema mono-objetivo con restricciones. La propuesta consiste en mutar a cada bacteria después del ciclo quimiotáxico usando el operador de generación de dirección de PSO que hace que la bacteria sea atraída hacia la mejor bacteria de la población. Para lo cual cada bacteria debe calcular su velocidad, tener memoria de las posiciones donde ha estado en cada ciclo quimiotáxico, y conocer la posición de la mejor bacteria de la población, parámetros básicos para el funcionamiento del operador del PSO. La propuesta fue probada en el problema de emisión de energía dinámico con restricciones, sin embargo los autores no mencionan como manejaron las restricciones del problema, los resultados obtenidos de dos casos del mismo problema (5 y 10 sistemas de unidades) se compararon con tres métodos encontrados en la literatura, de los cuales la propuesta fue quien obtuvo mejores resultados en minimizar el costo de la emisión de energía. Una desventaja de esta propuesta es el incremento de parámetros que el usuario tiene que definir (propios de PSO y BFOA).

Bakwad et al. [26] adaptan el proceso quimiotáxico y realizan en paralelo la evaluación de función objetivo del problema mono-objetivo sin restricciones en este proceso. La adaptación se hace presente al finalizar el proceso quimiotáxico, cuando se cambia la posición de cada bacteria agregando un componente social que es la bacteria con mejor posición de la población. La mejor posición es actualizada en cada iteración de este proceso, donde cada bacteria es evaluada paralelamente en la función objetivo. La propuesta fue probada para reducir el tiempo computacional de estimación de movimiento en la compresión de un video. No se reporta el número de ejecuciones independientes realizadas pero los resultados son comparados con 7 métodos encontrados en la literatura, los cuales son mejores según lo observado en sus gráficas y tablas reduciendo significativamente el tiempo de compresión de un video. Una característica relevante de esta propuesta es que se requieren varios computadoras, específicamente el mismo número que el de la población de bacterias. Sin embargo se puede ejecutar la propuesta de manera serial aunque el costo computacional para una sola computadora será más alto.

Hsiang-Chech et al. [53] adaptan el proceso quimiotáxico, añaden mutación y eliminan al proceso de eliminación-dispersión para resolver un problema mono-objetivo sin restricciones. En esta propuesta se adapta el proceso quimiotáxico en base a la diferencia de dos variables del problema para decidir hacia qué dirección debe moverse la bacteria, tomando en cuenta que la nueva dirección debe estar dentro de un rango determinado por éstas dos variables. La mutación es utilizada, donde algunos de los valores de las variables del diseño son alterados intencionalmente para dar oportunidad de escapar de óptimos locales. La propuesta fue probada en un problema que busca optimizar la imperceptibilidad y la robustez de las marcas de agua para proteger los datos de medios como audio, video e imágenes. Estos dos objetivos son formulados en una sola función objetivo colocando un parámetro de peso y para balancear estos dos objetivos se emplea teoría difusa. No se reporta el número de ejecuciones independientes realizadas a la propuesta pero se comparan resultados contra BFOA y un AG, donde la propuesta muestra mejores resultados. Cabe mencionar que esta propuesta adapta al BFOA exclusivamente para este problema y una de las desventajas relevantes es el sesgo de los resultados de acuerdo a las preferencias del usuario definido por el parámetro de peso.

Zhang et al. [54] utilizan BFOA para resolver un problema mono-objetivo sin restricciones donde se busca encontrar los valores de peso óptimos para entrenar una red neuronal destinada a la predicción de carga eléctrica a corto plazo, en dicha implementación no se modifica al algoritmo. Los resultados presentados son de una ejecución de BFOA con 2000 generaciones, los cuales son comparados con los resultados de un AG donde BFOA converge más rápido y realiza pronósticos más precisos.

Oyekan y Hu [55] utilizan el BFOA para resolver un problema mono-objetivo sin restricciones que consiste en encontrar los valores óptimos a las variables de diseño del controlador PID de un vehículo aéreo no tripulado. Los resultados obtenidos por BFOA son comparados con el método Ziegler Nicholas en los cuales BFOA presenta mejores resultados de acuerdo a las gráficas presentadas por los autores y converge más rápido en encontrar el óptimo. No se menciona el número de ejecuciones realizadas ni el valor de los parámetros usados en el algoritmo.

Chatterjee et al. [56] utilizan BFOA para resolver un problema mono-objetivo sin restricciones que busca optimizar las dimensiones físicas de los transistores PMOS y NMOS del CCII+. Este es un problema biobjetivo pero a través de una suma ponderada de objetivo el problema es modelado y resuelto como un mono-objetivo. El problema presenta una restricción detallada en el artículo pero no se menciona cómo esta restricción es evaluada en BFOA. Los resultados de BFOA son mejores comparados con PSO y ED, según las tablas presentadas por los autores, los cuales fueron generados en diferentes ejecuciones independientes aunque no se especifican cuantas.

Kumar y Jayabarathi [57] adaptan el tamaño de paso de BFOA para resolver un problema mono-objetivo sin restricciones. El tamaño de paso de las bacterias es actualizado al final de cada ciclo quimiotáxico por la diferencia del tamaño de paso de la bacteria en el ciclo quimiotáxico en proceso y el tamaño de paso al final del ciclo quimiotáxico, la cual es dividida entre la suma del número de pasos quimiotáxicos permitidos y el tamaño de paso al final del ciclo quimiotáxico. El tamaño de paso inicial es un valor resultante del producto de 0.05 por un valor entre 1 y el número de bacterias en la población. La propuesta es probada en 33 distintas configuraciones del problema donde se busca minimizar la perdida de suministro de energía eléctrica. Los resultados son comparados con diez algoritmos encontrados en la literatura que resuelven este mismo problema, entre los cuales está un AG, un hibrido de colonia de hormigas (ACO) y AG, entre otros. Sin embargo la propuesta de este trabajo es quien obtiene una mejor reducción de pérdida de energía. El problema presenta dos restricciones, sin embargo en este trabajo éstas no son detalladas, tampoco se muestra en el pseudocódigo presentado la forma de tratar estas restricciones con el algoritmo propuesto. Además se tienen más parámetros que el usuario tiene que definir como el valor 0.05 utilizado al definir el tamaño de paso inicial.

Luo y Chen [58] crean un hibrido entre BFOA y un algoritmo evolutivo elitista con mutación multi-padres (individuos) para afinar tres parámetros del controlador PID de un sistema servo eléctrico hidráulico de plataforma paralela 6-DOF, el cual es un problema mono-objetivo sin restricciones. En el proceso quimiotáxico de BFOA el movimiento de giro de una bacteria es realizado con el algoritmo evolutivo elitista, el cual se encarga de mutar bacterias de la población para generar la nueva posición de la bacteria en movimiento. Los resultados de la propuesta son simulados y de acuerdo a las gráficas presentadas el hibrido es una efectiva estrategia de afinación y obtiene buenos resultados en comparación con el método NN network y Input Curve. No se menciona el número de ejecuciones independientes realizadas a la propuesta. La mutación de bacterias es un proceso que requiere más costo computacional al evaluar cada mutación en la función objetivo del problema.

Biswas et al. [27] analizan el proceso de reproducción del BFOA y lo adaptan en su propuesta llamada ARBFOA para resolver problemas de optimización mono-objetivo sin restricciones. En esta propuesta la frecuencia de reproducción, determinada por el número de ciclos quimiotáxicos dentro de cada ciclo de reproducción, es aumentada gradualmente hasta el final del proceso. Para ello el número de ciclos quimiotáxicos es gradualmente reducido determinado por la diferencia del número de ciclo quimiotáxico inicial y el número del ciclo de reproducción en ejecución, multiplicado por el parámetro de cambio definido por el usuario donde los autores recomiendan que sea un valor entero positivo. La propuesta es probada en once funciones de prueba cada una con cincuenta ejecuciones independientes. Los resultados fueron comparados con el BFOA, donde la propuesta obtienen mejores resultados en ocho de las once funciones de prueba. El proceso de reproducción permite una convergencia más rápida en este tipo de problemas y la forma en que se adapta en esta propuesta es fácil de implementar, aunque requiere de un parámetro más, a los propios del BFOA definidos por el usuario.

Muñoz et al. [59] proponen simplificar al BFOA reduciendo el número de veces en la que se ejecuten los procesos de reproducción y eliminación-dispersión, la adaptación del tamaño de paso, la población inicial deja de ser aleatoria y se elimina la comunicación de bacteria a bacteria para resolver problemas de optimización sin restricciones. En la simplificación del BFOA se sustituyen los ciclos de reproducción y eliminación-dispersión por eventos esporádicos, es decir, cada cierto número de iteraciones se llevan a cabo estos procesos después de haber concluido el ciclo quimiotáxico, eliminando con esto el contador de movimientos permitidos a cada bacteria, el número de ciclos de reproducción y eliminación-dispersión. Para la adaptación del tamaño de paso se utiliza la regla de $\frac{1}{5}$ del algoritmo de Estrategias Evolutivas, donde el valor la variable de intensidad de mutación es incrementada en 1.20 si el valor de aptitud incrementa, es decir, no es una buena posición en el espacio de búsqueda, de lo contrario, el valor de la variable de intensidad es disminuida en 0.80. Cabe mencionar que esto es multiplicado por un factor de escalamiento R que es un valor aleatorio de distribución Gaussiana entre un rango definido por el usuario. Además se incluye la dirección de la posición de la mejor bacteria de la población a la fórmula de movimiento como guía de búsqueda en el proceso quimiotáxico. Otra de las modificaciones es que la población inicial del algoritmo es introducida de acuerdo al método de distribución uniforme Hammersley y por último, la eliminación de la comunicación de bacteria a bacteria. Esta propuesta fue probada con 30 ejecuciones independientes en 18 funciones de prueba. Los resultados fueron comparados con el original BFOA y otra propuesta derivada de BFOA encontrada en la literatura, donde el algoritmo propuesto obtiene mejores resultados en 4 problemas. En el algoritmo propuesto se elimina la variable de nado, sin embargo, sufre de convergencia prematura y en varias pruebas no se logra encontrar el mínimo global, lo cual es una desventaja al compararlo contra otras meta-heurísticas. Una de las posibles razones es la falta de elasticidad del tamaño de paso. En esta propuesta la población es inicializada con el método Hammersley lo cual requiere de un mínimo costo computacional y no permite una comparación justa contra otras meta-heurísticas donde su población inicial es aleatoria. Sin embargo esto no beneficia mucho al algoritmo ya que sus resultados no son competitivos. Cabe mencionar que éste no es el primer trabajo donde se simplifica el algoritmo del forrajeo de bacterias, ya que existe en la literatura el trabajo realizado por Mezura-Montes y Hernández-Ocaña [36].

Supriyono y Tokhi [60] adaptan el tamaño de paso para que este cambie dependiendo del valor de la función de aptitud encontrado por la bacteria para resolver problemas de optimización mono-objetivo sin restricciones. De este trabajo se derivan tres propuestas basadas en BFOA, donde las propuestas adaptan el tamaño de paso utilizando una función lineal, cuadrática y exponencial respectivamente. Cada una de las funciones utiliza una variable que contiene el valor máximo que puede tomar el tamaño de paso, un factor positivo, un factor escalar positivo y el valor de la función de aptitud por cada posición de la bacteria en el espacio de búsqueda. En cada una de las propuestas el tamaño

de paso será el producto del tamaño de paso por una de las funciones ya sea lineal, cuadrática o exponencial. Las propuestas son probadas con dos funciones de prueba y comparadas entre sí al igual que con el BFOA. Los autores no mencionan el número de ejecuciones independientes realizadas a las propuestas pero los resultados obtenidos por las tres propuestas son mejores que BFOA y se aproximan al óptimo global pero no se observa una diferencia significativa entre los resultados de las tres propuestas, ya que para el primer problema la propuesta con función cuadrática converge de manera más rápida comparado con los demás y en el segundo problema es la propuesta con función exponencial. Cabe mencionar que para ambos problemas se utilizaron valores diferentes en los parámetros del algoritmo. La implementación de las funciones lineal, cuadrática y exponencial es sencilla de realizar, sin embargo se agregan tres parámetros más que el usuario tiene que definir a parte de los propios del BFOA. Uno de los detalles sobre salientes es que los valores de los parámetros utilizados por los algoritmos propuestos son diferentes para cada problema de prueba debido a la dificultad de éstos según la opinión de los autores.

Ben et al. [61] proponen adaptar el tamaño de paso para que éste sea dinámico y así resolver problemas de optimización mono-objetivo sin restricciones. El Tamaño de paso es ajustado dinámicamente en cada iteración del algoritmo con el modelo de modulación decreciente no-lineal con el objetivo de balancear la exploración y explotación del espacio de búsqueda. Para este modelo se requieren cuatro nuevas variables: índice de modulación, coeficiente entre un rango de (0,7), límite superior y límite inferior de tamaño de paso. La propuesta es probada en dos funciones de prueba, no se menciona el número de ejecuciones independientes y la propuesta no es comparada con ninguna otra meta-heurística. Los resultados mostrados son de acuerdo a los distintos ajustes de las variables del modelo de modulación decreciente no-lineal. Donde los límites tanto superior como inferior del tamaño de paso deben ser valores inferiores a cero, logrando con esto una convergencia rápida y constante. Sin embargo la propuesta no logró llegar al óptimo global de los problemas probados. Una de las desventajas de esta propuesta es el incremento de parámetros (cuatro) a definir por el usuario.

Chen et al. [62] autoadaptan el tamaño de paso utilizado en el proceso quimiotáxico para resolver un problema multi-objetivo sin restricciones, dichas funciones objetivo fueron sumadas para resolver el problema como un mono-objetivo sin restricciones. La autoadaptación del tamaño de paso se realiza con el propósito de explorar el área de búsqueda con un tamaño de paso grande ya sea para nadar o girar, y de explotar la región en la cual se localiza la bacteria usando un tamaño de paso menor al moverse (nadar o girar). El mecanismo funciona con dos criterios de estados realizados de manera paralela para cada bacteria, 1) si la bacteria descubre una región prometedora, es decir, con presencia de nutrientes donde su valor de aptitud es mejor al previo, el tamaño de paso de la bacteria decrece al dividir el tamaño de paso actual entre una constante definida por el usuario por cada iteración del proceso quimiotáxico, donde la mejora es medida con un parámetro de precisión, el cual es definido por el usuario y es modificado en las iteraciones del proceso quimiotáxico al dividir la precisión que la bacteria tiene entre una constante definida por el usuario. 2) Si después de un número de generaciones determinado por el usuario, la bacteria no mejora su búsqueda, entonces el tamaño de paso aumenta, es decir, vuelve a tomar el valor del tamaño de paso inicial definido por el usuario y lo mismo aplica para el parámetro de precisión. Si no se presenta ninguno de los dos criterios de estado, entonces el tamaño de paso y la precisión siguen con los mismos valores obtenidos hasta ese momento en la ejecución. La propuesta es probada para optimizar la red dinámica RFID la cual tiene dos funciones objetivo, optimizar la cobertura de etiquetas y optimizar el número de lectores de prevención de colisiones. No se menciona el número de ejecuciones independientes realizadas pero los resultados obtenidos por la propuesta son mejores comparados con los del BFOA. Sin embargo, una desventaja de esta propuesta es la integración de cinco parámetros más al algoritmo, tres de ellos son inicializados y los otros dos son definidos por el usuario (parámetro usado en el criterio 2).

Tabatabaei y Vahidi [63] proponen usar el BFOA para asistir el método de lógica difusa que resuelve un problema particular de optimización mono-objetivo sin restricciones que corresponde a encontrar la óptima ubicación y tamaño de derivación de capacitores en el sistema de distribución radial. Del problema se generan dos casos en los cuales se prueba la lógica difusa asistida por BFOA. Los resultados obtenidos son mejores comparados con la lógica difusa normal sin embargo no se menciona el número de ejecuciones independientes.

Sanyal et al. [64] utilizan SA-BFO una propuesta derivada de BFOA [62] autoadaptando el tamaño de paso de acuerdo a dos criterios de estados basados en el valor de aptitud, los detalles de esta propuesta son descritos en [62]. SA-BFO es aplicado para optimizar la entropía borrosa o difusa en la segmentación de imágenes en gris. Se realizaron veinte ejecuciones independientes de SA-BFO y los resultados son comparados con BFOA donde la propuesta minimiza mucho más la entropía difusa de las imágenes tratadas. Una de las desventajas de esta propuesta son los parámetros que usan estos criterios de estado los cuales son definidos por el usuario.

Sathya y Kayalvizhi [65] implementan BFOA para resolver el problema mono-objetivo sin restricciones que busca encontrar el óptimo valor de alumbramiento en la segmentación de imagen utilizando las técnicas de alumbramiento llamadas Kapur y Otsu. Se obtuvieron resultados para ambos métodos (Kapur y Otsu) de 10 imágenes diferentes con cuatro niveles de alumbramiento, donde los resultados que obtuvo BFOA fueron mejores comparado con los resultados de PSO y AG. En el método Kapur BFOA obtuvo una desviación estándar menor que los otros algoritmos y en el método Otsu, BFOA convergió rápidamente.

Sathya y Kayalvizhi [66] modifican el tamaño de paso de BFOA para resolver un problema mono-objetivo sin restricciones. El tamaño de paso en esta propuesta es adaptable para cada bacteria y es calculado usando la división del valor de aptitud de la bacteria en proceso entre éste mismo valor sumado a una constante positiva definida por el usuario. La propuesta es aplicada para maximizar dos funciones objetivo por separado, la primera es un método llamado Kapur y la segunda el método Otsu, las cuales son dos técnicas populares de alumbramiento para la segmentación de imágenes de resonancia magnéticas del cerebro. Se obtuvieron resultados para ambos métodos (Kapur y Otsu) de 10 segmentaciones con cuatro niveles de alumbramiento, donde los resultados de la propuesta fueron mejores comparado con los resultados de BFOA, PSO y AG, según lo presentado en tablas por los autores. La adaptación de tamaño de paso en esta propuesta es sencilla de implementar, sin embargo, se requiere de un parámetro más a los propios de BFOA a definir por el usuario.

Estos mismos autores Sathya y Kayalvizhi [67] modifican el proceso de agrupamiento y reproducción del BFOA para resolver un problema mono-objetivo sin restricciones. En el proceso de agrupamiento la posición de todas las bacterias en un nuevo ciclo quimiotáxico son evaluadas para encontrar a la mejor bacteria global. En el proceso quimiotáxico cada bacteria nada de acuerdo a la suma de la posición actual más la posición actual de la bacteria global y sí es giro, el movimiento es aleatorio. El proceso de reproducción a diferencia de BFOA donde la salud de una bacteria es determinada por el valor promedio de los valores de aptitud obtenidos por la bacteria en el proceso quimiotáxico, es decir, es el máximo valor de aptitud (valor de la función objetivo) de todos los ciclos quimiotáxicos de la bacteria en proceso. Después las bacterias con mejor salud son duplicadas. La propuesta es empleada para encontrar el valor óptimo de alumbramiento en la segmentación de imagen con dos técnicas de alumbramiento llamados método Kapur y método Otsu. Se obtuvieron resultados para ambos métodos de 14 imágenes diferentes con cuatro niveles de alumbramiento, donde los resultados que obtuvo la propuesta fueron mejores comparado con los resultados de BFOA, PSO y AG según

lo presentado en tablas e imágenes por los autores. Sin duda BFOA es un algoritmo que obtiene resultados competitivos en este tipo de problemas de segmentación de imágenes.

Sharma y Kanaujia [68] implementan BFOA para resolver un problema mono-objetivo sin restricciones que consiste en optimizar el cálculo de la frecuencia resonante de una microcinta circular de una antena con o sin espacios de aire. Los resultados obtenidos fueron comparados con tres modelos encontrados en la literatura, donde BFOA obtiene mejores resultados de acuerdo a las tablas y gráficas presentadas por los autores pero no se menciona el número de ejecuciones independientes realizadas.

Lei et al. [69] implementan BFOA para resolver un problema mono-objetivo sin restricciones que busca pronosticar los módulos funcionales de una red PPI, para ello se adaptó el BFOA a las características particulares del problema, donde los resultados obtenidos fueron comparados con el algoritmo Flow encontrado en la literatura, pero es BFOA quien obtiene mejores resultados para la función objetivo del problema. No se menciona el número de ejecuciones independientes realizadas al algoritmo.

Verma et al. [70] implementan BFOA con cambios en la forma en que las bacterias realizan el movimiento de nado giro para resolver un problema mono-objetivo sin restricciones. La propuesta consiste en calcular la dirección de giro de la bacteria en base a la técnica de probabilidad derivada la cual es basada en posibles direcciones que puede tomar la bacteria definidas en una matriz que contiene las direcciones norte, sur, este, oeste, suroeste, noroeste, sureste y noreste. La propuesta es probada para la detección de contorno de imágenes, donde las bacterias tienen que encontrar los pixeles de interés de la imagen, no se menciona el número de ejecuciones independientes pero se muestran los resultados de cinco imágenes aplicando BFOA los cuales son comparados con los detectores de contorno Sobel, Canny, Edison, Rothwell y SUSAN encontrados en la literatura. Las imágenes resultantes y tablas muestran que BFOA es un algoritmo que permite detectar contornos pero no exitosamente como los detectores con los que fue comparado. Una de las desventajas de esta propuesta es la definición de los valores en la matriz de direcciones.

Patnaik y Panda [71] implementan el BFOA para resolver un problema mono-objetivo sin restricciones donde se busca optimizar la corriente armónica mediante el empleo de filtro activo de potencia. Los resultados de BFOA son comparados con PSO, la sintonización convencional del controlador PI basado en filtro activo de potencia y el convencional PI. De acuerdo a las gráficas presentadas BFOA obtiene los mejores resultados para la función objetivo de este problema. No se menciona el número de ejecuciones realizadas al algoritmo.

Kamyab y Bahrololoum [72] implementan al BFOA cambiando el proceso de eliminación-dispersión para resolver un problema mono-objetivo sin restricciones. En el proceso de eliminación-dispersión se usa el operador de PSO para mutar la posición y velocidad de la bacteria que será eliminada y dispersar esta bacteria nuevamente con los valores resultantes del operador de PSO. La propuesta fue probada en el problema de afinación de parámetros del sistema difuso tipo TKS basado en reglas. Los resultados de la propuesta fueron comparados con PSO y otros dos algoritmos encontrados en la literatura, donde la propuesta obtiene resultados competitivos y en algunos casos mejores. Cabe mencionar que la implementación del PSO en la propuesta es fácil de entender y realizar, sin embargo se agregan cuatro parámetros que hacen funcionar al operador PSO que son definidos por el usuario. En este trabajo no se detalla claramente el resultado del problema, simplemente se muestran dos gráficas del comportamiento de los algoritmos comparados, tampoco se menciona el número de ejecuciones independientes realizadas para probar dicha propuesta.

Supriyono y Tokhi [73] adaptan el tamaño de paso del proceso quimiotáxico del BFOA para resolver un problema mono-objetivo sin restricciones. El tamaño de paso en esta propuesta es calculado usando una estructura de lógica difusa construida con 7 reglas a la cual es introducido el valor absoluto del costo de la función objetivo. El objetivo de la estructura es determinar el valor del tamaño de paso, sí el valor de aptitud de la bacteria en proceso es grande entonces el tamaño de paso es grande, de lo contrario es pequeño. El algoritmo propuesto funciona como algoritmo de aprendizaje para optimizar los parámetros de una red neuronal para modelar un sistema manipulador flexible de un solo enlace. No se menciona el número de ejecuciones independiente realizadas a la propuesta pero los resultados obtenidos fueron comparados con BFOA y fue el algoritmo propuesto quien obtiene mejores valores, menor error en la precisión de los parámetros y una rápida convergencia. La adaptación del tamaño de paso es sencilla, sin embargo, la definición de las reglas de la estructura deben ser modeladas de acuerdo a las características del problema y necesidades del usuario. El rango del tamaño de paso en esta propuesta varía entre [0.0019, 0.046].

Abd-Elazim y Ali [74] implementan BFOA para resolver un problema mono-objetivo sin restricciones que consiste en ajustar los parámetros del controlador SVC que es un compensador var estático para la supresión de oscilaciones en un sistema de energía multi-máquinas. No se mencionan el número de ejecuciones independientes realizadas al algoritmo pero los resultados mostrados en las gráficas y tablas muestran que BFOA obtiene mejores resultados al minimizar la función objetivo del problema por debajo de los resultados obtenidos por dos algoritmos encontrados en la literatura. También la convergencia de BFOA fue más rápida.

Vivekanandan y Ramyachitra [75] implementan BFOA para resolver un problema mono-objetivo sin restricciones que consiste en buscar secuencias de proteínas similares en distintas bases de datos en un ambiente grid. El problema no es detallado, tampoco se menciona el número de ejecuciones independientes realizadas a BFOA, sin embargo, se presentan resultados que son comparados con dos algoritmos encontrados en la literatura, donde BFOA, en la mayoría de los casos presentados, obtiene mejores resultados.

Nouri y Hong [76] implementan BFOA para resolver un problema de optimización mono-objetivo sin restricciones, donde la forma de realizar los movimientos de nado-giro de las bacterias es adaptado de acuerdo al problema a resolver. El problema que se resuelve en este trabajo es una estrategia para agrupar tipos de producción llamado sistema de fabricación celular. El objetivo es agrupar la producción de máquinas con características similares. Este problema es representado con una matriz, donde las filas corresponden al tipo de máquina y las columnas a las características diferentes de cada tipo de máquina llamadas celdas. El problema de optimización consiste en minimizar el número de celdas vacías, donde cada celda puede tener como mínimo 2 tipos de máquina. Debido a esta representación de matriz, el movimiento de nado-giro de una bacteria es en base a la matriz, donde los autores proponen una mutación global o local, es decir, sí en el nado o giro de la bacteria el valor de una variable llamada probabilidad de mutación (PGMT) es mayor al valor de una variable aleatoria llamada RND entonces se aplica mutación global que consiste en cambiar los valores de una columna (celda) a otra columna de la matriz. En el caso de que la PGMT es menor a RND entonces se aplica mutación local que consiste en cambiar los valores de una fila a otra en la matriz. Para el caso de la reproducción de bacterias, los autores proponen que el número de bacterias a reproducir sea solo el 20 % de las mejores bacterias de la población. El algoritmo propuesto fue probado en 46 funciones de prueba encontradas en la literatura con diferentes dimensiones de matriz, no se menciona el número de ejecuciones independientes. Los resultados de estas pruebas fueron comparados con los resultados de once algoritmos encontrados en la literatura, donde los resultados del algoritmo propuesto son mejores de acuerdo a las tablas presentadas por los autores. La propuesta logra una eficacia en el

agrupamiento de las máquinas, sin embargo, no se menciona las consecuencias de la mutación loca o global al cambiar de manera total una máquina de su tipo o que sus características cambien por completo. Por otra parte, un nuevo parámetro es requerido PGMT que es definido por el usuario, además esta propuesta es aplicable sólo a este problema ya que sería difícil implementarlo para otro problema debido a la matriz utilizada.

Kushwaha et.al., [77] proponen un híbrido entre un GA basado en BFOA, donde se aplican los operadores de mutación y cruce a las bacterias en el proceso quimiotáxico, después de los movimientos de nado-giro.

2.1.2. Optimización global con restricciones basada en BFOA

Mezura-Montes y Hernández-Ocaña [78] proponen el BFOA modificado (MBFOA) para resolver problemas mono-objetivo con restricciones, particularmente de diseño mecánico. Las modificaciones pueden dividirse en dos partes: (1) la simplificación del original BFOA (disminución del número de parámetros definidos por el usuario y eliminación de uno de los cuatro ciclos anidados *for*) y (2) uso de una técnica para el manejo de restricciones. Como resultado, en MBFOA cada bacteria realizado su propio ciclo quimiotáxico dentro de un ciclo de generación. Los ciclos de reproducción y eliminación-dispersión se han cambiado como pasos individuales. Por otra parte, el valor de tamaño de paso es definido mediante una fórmula basada en los límites de las variables del diseño. Un mecanismo de agrupamiento también se añadió para redefinir el paso quimiotáxico, es decir, a la mitad y al final del ciclo quimiotáxico, en lugar de un movimiento de nado o giro, cada bacteria es atraído por la mejor bacteria de la población. Se hace uso de las reglas de factibilidad de Deb para el manejo de las restricciones (técnica de segunda generación) [37]. La propuesta fue probada en tres conocidos problemas de diseño de ingeniería y los resultados obtenidos se compararon frente a variantes de PSO donde MBFOA proporcionó resultados competitivos con un menor número de evaluaciones. Las principales ventajas de MBFOA es la reducción de los parámetros definidos por el usuario y un ciclo interno de BFOA. Sin embargo, el enfoque fue particularmente sensible a el valor de tamaño de paso, aunque se definió utilizando los límites de las variables de decisión. Por último, las reglas de factibilidad permiten a MBFOA alcanzar consistentemente la región factible del espacio de búsqueda, pero su fuerte preferencia por soluciones factibles llevaron a convergencia prematura en algunas ejecuciones.

Praveena et al. [79] proponen un hibrido entre el algoritmo ED, PSO y BFOA para resolver un problema mono-objetivo con restricciones. La propuesta consiste en incluir a BFOA el operador de generación de dirección de PSO donde se definen, almacenan y actualizan la mejor posición de la bacteria y la mejor posición global de todas las bacterias en la población en una iteración del ciclo quimiotáxico. Para los movimientos de la bacteria, nado o giro, el vector de direcciones aleatorio es reemplazado por la velocidad de la bacteria que se obtiene con la fórmula de PSO. Después de los movimientos de la bacteria se hace presente ED de acuerdo a una probabilidad de presencia para cruzar a la bacteria con dos individuos distintos de la población. Cabe mencionar que en cada proceso (iniciación, quimotáxico, cruza) el manejo de las restricciones del problema fue con la función de penalización una técnica de primera generación. El hibrido propuesto fue implementado para resolver el problema de emisión de energía con efectos de punto-válvula. Los resultados obtenidos son mejores al obtener un mejor costo para la función objetivo comparado con algoritmos encontrados en la literatura. No se menciona el número de ejecuciones independientes realizadas al hibrido. Una de las desventajas de la propuesta son el aumento de parámetros a definir por el usuario debido a la incorporación de los mecanismos principales de PSO y ED.

Chunguang et al. [80] implementan BFOA para resolver un problema mono-objetivo con restricciones que consiste en minimizar la perdida de ingredientes en la dilución de cobre. Las restricciones del problema son manejadas con una función de penalización (técnica de primera generación) basada en el método Lagrange relax. Los resultados obtenidos por BFOA son mejores en comparación que el método Experience Heuristic al encontrar en 5 ejecuciones independientes valores menores en el costo de la función objetivo del problema. La ventaja de esta propuesta son los resultados obtenidos y la simplicidad de la técnica de manejo de restricción utilizada, sin embargo los parámetros de BFOA y los factores de penalización fueron definidos por el usuario, los valores a estos parámetros no se reportan en los experimentos.

Kou et al. [81] implementan BFOA para resolver un problema mono-objetivo con restricciones que consiste en el estudio de la corriente térmica de pruebas no-destructiva y evaluación (TNDT/E) para identificar parámetros con defectos. Este problema presenta restricciones las cuales son tratadas con la técnica de manejo de restricción llamada penalización de muerte (técnica de primera generación) basada en el modelo sustituto RBFNN, el cual es una combinación del método de superficie de respuesta, modelo Kriging, red neuronal y la función de base radial. No se mencionan el número de ejecuciones independiente tampoco tablas estadísticas, sólo se presentan gráficas de convergencia con los resultados obtenidos por BFOA en los que se menciona que son mejores que PSO al minimizar la función objetivo de manera más rápida según estas gráficas.

Pangrahi et al. [82] adapta el proceso quimiotáxico, elimina el proceso de reproducción para resolver un problema multi-objetivo con restricciones. En esta propuesta se adapta la forma en que debe girar la bacteria ya que el giro puede ser completamente aleatorio o de acuerdo a una probabilidad llamada guía, en la cual la bacteria gira hacia la dirección previa. La dominancia difusa es integrada al proceso quimiotáxico para seleccionar las mejores bacterias que conformarán la población para la siguiente iteración de este proceso. El manejo de las restricciones es realizado por los métodos *non-dominated sorting* y *fuzzy dominance sorting* los cuales son técnicas para el manejo de restricciones utilizadas en el concepto multi-objetivo de segunda generación. La propuesta es probada en el problema que busca minimizar el costo de la emisión de despacho de carga eléctrica. Se realizaron 15 ejecuciones independientes a la propuesta y se comparan sus resultados con seis algoritmos encontrados en la literatura. La propuesta es quien obtiene los mejores resultados al problema obteniendo un frente de Pareto competitivo y diverso con el uso del método de aproximación difusa. Esta propuesta agrega otros parámetros a definir por el usuario.

Estos mismo autores resuelven el mismo problema en el 2011 [83], utilizando la técnica de dominancia de Pareto como criterio de selección y adaptando de nuevo el proceso quimiotáxico. En el proceso quimiotáxico cuando la bacteria decide girar debido a que no se encuentra en una región o espacio favorable para su supervivencia puede hacerlo aleatoriamente hacia cualquier dirección o dirigirse hacia la posición de una de las bacterias que tienen mejor *Rank* (bacteria que este menos dominada por otra, *Rank* obtenido con la técnica de dominancia de Pareto) según una probabilidad llamada guía y sí la bacteria en proceso tiene su *Rank* mayor, es decir, es dominada más de una vez. Con el uso de la técnica de dominancia de Pareto como criterio de selección solo se seleccionan las bacterias con mejor *Rank*, es decir, aquellas bacterias que son menos dominadas por otras de la población para ser las que permanezcan para la siguiente iteración de proceso quimiotáxico. Se realizaron 15 ejecuciones independientes a la propuesta y se comparan sus resultados con ocho algoritmos encontrados en la literatura. Los resultados obtenidos por la propuesta son competitivos ya que se encuentran dentro del frente óptimo de Pareto obtenido mediante el método de ϵ-Constraint. Sin embargo, los demás algoritmos con los que se comparó también obtienen resultados similares de acuerdo a los frentes de

Pareto obtenidos en cada caso probado. Esta propuesta es fácil de implementar para resolver otro tipo de problemas multi-objetivo con restricciones, sin embargo se agrega el parámetro probabilidad de guía que es definido por el usuario.

Deshpande et al. [84] modifican al BFOA para resolver un problema multi-objetivo con restricciones. La propuesta consiste en utilizar meta-programación difusa para formular el problema multi-objetivo con restricciones a resolver de acuerdo a las aspiraciones que el usuario desea obtener, para la toma de decisiones, colocando una variable de peso a cada una de las funciones objetivo. La técnica de manejo de restricción usada fue penalización de muerte que pertenece a la primera generación derivada de la función de penalización. Cuando un vector de solución viola restricciones entonces es generado un nuevo vector de solución aleatoriamente hasta que no se viole ninguna restricción. Además se ha incorporado un mecanismo a BFOA que permita que el algoritmo incorpore variables enteras como variables de optimización, donde el vector inicial generado es validado para que sus elementos sean valores enteros al igual que el vector direccional de las bacterias. Es decir, la dirección y el tamaño de paso de cada bacteria es redondeado al entero más próximo para mantener la factibilidad de la solución del problema. El problema multi-objetivo con restricciones consiste en llevar un control del inventario de productos en una cadena de suministros, que consta de tres funciones, minimizar los costos de operación total, minimizar el tiempo de entrega del producto y maximizar el nivel del índice de servicio, todas estas bajo un conjunto de restricciones. El algoritmo propuesto es probado en dos casos diferentes, el primero en donde el producto es único en el mercado y el cliente puede esperar su abastecimiento, y el segundo donde existen otros productos similares en el mercado donde los clientes fácilmente pueden comprar otro sin necesidad de esperar el abastecimiento. En ambos caso la meta-programación difusa es quien debe ponderar las funciones objetivo del problema de acuerdo al nivel de aspiraciones del usuario. El algoritmo propuesto obtiene mejores resultados de acuerdo a las gráficas presentadas por los autores satisfaciendo cada una de las restricciones y optimizando las funciones objetivo del problema. Sin embargo, es importante definir cuidadosamente la meta-programación difusa para que BFOA obtenga resultados eficientes para el tomador de decisiones. Otra de las características de esta propuesta es que el número de parámetros a definir por el usuario aumenta, ya que la meta-programación difusa tiene sus propios parámetros. El costo computacional también incrementa debido a que el manejo de las restricciones es a prueba y error, es decir, un vector de soluciones se evalúa en las funciones objetivo hasta que éste no viola ninguna restricción.

Mezura-Montes y López-Dávila [34] adaptan el tamaño de paso, reducen la presencia del proceso de reproducción y eliminación-dispersión e implementan un operador de búsqueda local en el proceso quimiotáxico para resolver problemas mono-objetivo con restricciones. El tamaño de paso es disminuido de acuerdo al comportamiento exitoso de los movimientos (nados-giros) de la bacteria en un ciclo del proceso quimiotáxico, donde sí el promedio de los movimientos exitosos es menor a 0.2 entonces el nuevo tamaño de paso es definido por el nuevo tamaño de paso en la generación actual multiplicado por un parámetro de ajuste inspirado en la regla de $\frac{1}{5}$ del algoritmo de Estrategia Evolutiva para controlar el nivel de cambio calibrado por el usuario. De otra manera, sí el promedio de movimientos exitosos es igual o mayor a 0.2, el nuevo tamaño de paso es definido por el tamaño de paso de la generación anterior multiplicado por la división de uno entre el parámetro de ajuste inspirado en la regla de $\frac{1}{5}$. El tamaño de paso inicial del algoritmo es definido por la multiplicación de un parámetro de porcentaje definido por el usuario y la diferencia del límite superior e inferior de las variables del problema a resolver. La técnica de manejo de restricciones utilizada son las reglas de factibilidad de Deb [37], técnica de segunda generación. Además un método de programación matemática llamado Hooke-Jeeves fue adoptado como buscador local. Este método fue aplicado a un porcentaje de bacterias definido por el usuario (10 % de las mejores bacterias de la población) de acuerdo a las reglas de factibilidad. Las bacterias obtenidas por Hooke-Jeeves reemplazan

a las peores bacterias de la población. Se disminuye la presencia de los procesos de reproducción y eliminación-dispersión para favorecer la convergencia del algoritmo. Se realizaron 30 ejecuciones independientes a la propuesta en cada uno de los trece funciones de prueba que se utilizaron para probar esta propuesta. Los resultados fueron comparados con el MBFOA original y una subversión de la propuesta, donde ésta no implementa el operador de búsqueda local sólo la adaptación del tamaño de paso. Los resultados muestran que la propuesta obtiene mejores resultados que las otras dos meta-heurísticas. Los resultados de la subversión no tienen una mejora significativa con respecto a MBFOA. Los resultados de la propuesta son comparados contra 6 algoritmos encontrados en la literatura, donde el algoritmo propuesto presenta competitividad, sin embargo fue atrapado en un óptimo local en dos de los trece problemas de prueba. Además el costo computacional aumenta al realizar la búsqueda local Hooke-Jeeves. Los autores mencionan que el factor de escalamiento y el parámetro de ajuste inspirado en la regla de $\frac{1}{5}$ son muy sensibles en cada uno de los problemas probados, en particular el factor de escalamiento requiere casi valores opuestos (altos o bajos) en los diferentes tipos de problemas de prueba. Sin duda alguna, éste es un parámetro que el usuario tardará en ajustar debido a su sensibilidad. Generalmente esta propuesta presentan dos desventajas particulares, aumento en el costo computacional y el número de parámetros a definir por el usuario.

Pandit et al. [85] adaptan el tamaño de paso del proceso quimiotáxico, utilizan un operador de mutación para generar una nueva bacteria y eliminan el proceso de reproducción para resolver un problema mono-objetivo con restricciones. Una de las características de esta propuesta es que su población es de solo tres bacterias donde su tamaño de paso es dinámico haciendo uso de un repetidor llamado mapa logístico el cual permite variar el tamaño de paso entre el rango [0,1]. Al final del ciclo de nado-giro del proceso quimiotáxico, se localiza la mejor bacteria, de acuerdo a un *rank* en base al valor de la función objetivo, y conserva su posición, a la segunda bacteria se aplica la operación de mutación, de acuerdo a una probabilidad definida por el usuario, donde muta sus variables de diseño aleatoriamente, es decir, el valor de cada variable de diseño es el producto de un valor aleatorio por el valor de la variable de diseño actual. Y por último la peor bacteria es eliminada y reinicializada en una nueva posición. La técnica de manejo de restricciones utilizada fue función de penalización de la primera generación. La propuesta derivada de BFOA es probada en el problema multi-objetivo con restricciones que busca minimizar el costo y emisión de energía solicitada en ciertos periodos de tiempos. Dicho problema fue resuelto como mono-objetivo con restricciones. Se realizaron pruebas en cuatro casos de dicho problema y los resultados fueron comparados con siete algoritmos encontrados en la literatura incluido BFOA. Los resultados de la propuesta fueron los mejores aunque no se menciona el número de ejecuciones independientes realizadas. Esta propuesta presenta una población pequeña pero efectiva para este problema en particular, la propuesta de mutación es fácil de entender e implementar, sin embargo, se introduce un nuevo parámetro definido por el usuario para determinar la probabilidad de mutación. Por otra parte, el tamaño de paso es derivado del llamado mapa logístico que es un generador de numérico con secuencias caóticas utilizado en algoritmos evolutivos.

Vaisakh et al. [86] crean un hibrido de BFOA, PSO y Evolución Diferencial (ED) para resolver un problema mono-objetivo con restricciones, reemplazando la forma en que se generan las direcciones de movimientos de las bacterias, adaptando el tamaño de paso y utilizando mutación. La propuesta consiste en reemplazar la fórmula de generación de dirección aleatoria utilizada para direccionar junto con el tamaño de paso, el movimiento de nado o nado-giro de la bacteria. Donde esta fórmula es reemplazada por la generación de velocidad de PSO. El tamaño de paso es un valor aleatorio entre 0.1 y 5. Después del ciclo de nado de las bacterias, de acuerdo a una probabilidad definida por el usuario, hace presencia ED, donde se mueven a las bacterias hacia nuevas posiciones utilizando el operador de ED (cruza y mutación) siempre y cuando mejore el valor de aptitud de la bacteria

y la violación de restricciones sea menor o nula. La técnica para el manejo de restricciones fue una función de penalización de primera generación. Una vez realizado el ED en las bacterias, entonces se procede a actualizar por cada bacteria su mejor posición hasta el momento (las bacterias guardan en memoria todos sus movimientos en cada iteración del ciclo de nado-giro) y también el valor de la mejor posición global de la población es actualizado. En la iniciación del algoritmo la mejor posición de la bacteria es la misma generada aleatoriamente y para generar la mejor posición global, se evalúan todas las bacterias de la población en la función objetivo y restricciones, aquella con mejor valor de aptitud y menor violación de restricciones es considerada la mejor posición global. Ambas posiciones son actualizadas al final dentro del proceso quimiotáxico. La propuesta fue probada para minimizar el costo de la emisión de energía en ciertos periodos de tiempo con varias restricciones. Se realizaron diferentes casos del mismo problema y los resultados fueron comparados contra seis algoritmos encontrados en la literatura donde la propuesta obtiene mejores resultados.
En este hibrido el tamaño de paso ya no es calibrado por el usuario ya que es un valor aleatorio entre un rango determinado por los autores. Sin embargo se agregan más parámetros utilizados propiamente en la fórmula de PSO y ED que tienen que ser definido por el usuario, por ejemplo en PSO, constantes positivas como coeficientes de aceleración, factor de inercia. En el caso de ED, factor de amplificación de variación diferencial. Esta propuesta de hibrido no es el único encontrado en la literatura, ya que [79] también proponen un hibrido similar.

Saber [87] incluye el proceso quimiotáxico, en específico al movimiento de nado de una bacteria y el tamaño de paso, al algoritmo PSO para resolver un problema mono-objetivo con restricciones. Al operador PSO, que consiste en mutar la posición y velocidad de un individuo hacía la dirección del líder, a este proceso se agrega un parámetro de forrajeo de ubicación de paso aleatorio que consiste en añadir al individuo en proceso la posición del mejor individuo global (solo cierto porcentaje), siempre y cuando, el individuo en proceso no sea el mismo que el mejor individuo global, sí esto ocurre, entonces el parámetro forrajeo de ubicación de paso es completamente aleatorio. A la velocidad del individuo es agregado el tamaño de paso, donde este se calcula con la sumatoria de error (manejo de restricciones, entre menor sea el error más factible es la solución) dividido entre el número de iteración del PSO elevado al cuadrado. La técnica de manejo de restricciones utilizada es función de penalización de la primera generación. La propuesta fue aplicada al problema de emisión de energía donde se busca minimizar el costo de generación total de energía sujeto a restricciones de igualdad y desigualdad. Los resultados obtenidos con la propuesta fueron mejores, con respecto a dos algoritmos encontrados en la literatura con los que se comparó, según las tablas mostradas por los autores. No se menciona el número de ejecuciones independientes realizadas pero la ventaja de esta propuesta es la facilidad de obtener mejores resultados con menos costo computacional además de utilizar la información factible como control del tamaño de paso en la búsqueda, sin embargo los parámetros que el usuario tiene que definir son varios entre los que están los propios de BFOA, PSO y los factores de penalización.

Niu et al. [88] modifican el cálculo del tamaño de paso de BFOA para resolver un problema mono-objetivo con restricciones. El cálculo del tamaño de paso en esta propuesta se realiza durante el proceso de reproducción y eliminación-dispersión el cual consiste en disminuir el tamaño de paso inicial en un máximo número de iteraciones, usando un tamaño de paso inicial pequeño, un tamaño de paso final grande y el número máximo de ciclos del proceso quimiotáxico combinados en una fórmula de adición, diferencias y divisiones. Cuando el tamaño de paso inicial y final son iguales, el tamaño de paso es calculado con la fórmula original de BFOA. La propuesta fue probada en cuatro funciones de prueba unimodal y multimodales, los resultados obtenidos en cada función son mejores en comparación con BFOA. De igual manera, la propuesta fue implementada para resolver el problema de optimización de portafolios con riesgos de liquidez, las dos restricciones del problema son

tratadas en el algoritmo por la técnica de primera generación función de penalización. Los resultados de la propuesta en este problema son mejores que los resultados de los algoritmos BFOA, PSO y AG los cuales son mostrados en las tablas de estadísticas. También en la gráfica de convergencia se observa que la propuesta y BFOA realizan una convergencia rápida y similar la cual es mucho mejor que PSO y AG. Sin duda la propuesta obtiene excelentes resultados, sin embargo, son agregados dos parámetros más que el usuario tiene que definir aunque la implementación del cálculo del tamaño de paso es sencilla.

MBFOA [78] fue extendido por Mezura-Montes et al. [89] para resolver un problema de optimización de diseño mecánico multi-objetivo de una transmisión de variación continua, en donde la salida de un mecanismo de cuatro barras debe ser maximizada mientras que las diferencias de los ángulos de transmisión deben ser minimizados. Además de restricciones de igualdad y desigualdad, el problema multi-objetivo considera en su modelo una restricción de desigualdad dinámica relacionado con un ángulo que debe mantener su valor por debajo de un valor específico a lo largo de un ciclo de arranque. Dominancia de Pareto junto con las reglas de factibilidad de Deb [37] (técnica de segunda generación) se adoptó como criterio de selección en la propuesta. Se añadió distancia de aglomeramiento (*Crowding distance*) como mecanismo de diversidad en el espacio objetivo, elitismo a través de un archivo externo y la reproducción de bacterias fue mínima. Los resultados obtenidos se compararon con tres algoritmos encontrados en la literatura donde MBFOA obtuvo resultados competitivos. La principal ventaja de la propuesta fue que el frente de Pareto obtenido era claramente diferente de los alcanzados por los demás algoritmos, es decir, obtuvo muestras diferentes (y más regiones adecuadas para el problema) en el espacio de búsqueda. Por otra parte, se llegó constantemente a la región factible del espacio de búsqueda, incluso en presencia de la restricción dinámica. De hecho, se encontraron soluciones factibles en generaciones tempranas. Sin embargo, el algoritmo fue sensible a sus parámetros, en particular el valor de tamaño de paso. Por otra parte, el desempeño basado en indicadores como *two-set coverage* y *hypervolumen* no fue tan competitivo.

2.1.3. Análisis de la optimización global basada en BFOA

El presente análisis es referente a la revisión del estado del arte sobre la implementación de BFOA en problemas de optimización numérica, haciendo énfasis en problemas con presencia de restricciones. En la Tabla 2.1.3 se resumen las principales características de cada propuesta de BFOA usando alguna técnica de manejo de restricción.

- BFOA ha sido aplicado a diferentes problemas particulares donde destacan el ajuste de parámetros para sistemas como red PDI, manipulador flexible, sistemas de transmisión de variación continua entre otros más [73, 89, 44, 43, 18, 21, 32, 55, 71, 54, 63, 74, 56, 72, 58, 69], la predicción de emisión de energía [85, 52, 57, 79, 82, 83], tratamiento de imágenes y videos [50, 64, 67, 65, 66, 70, 53, 26], problemas relacionados con ajuste de señales de redes y antenas [46, 62, 49, 69, 68] y problemas del benchmark [61, 36, 13, 30, 31, 14, 17, 90, 51, 14, 27, 59, 60, 36, 34]. En algunos problemas particulares se hace presencia de dos objetivos, sin embargo algunas propuestas basadas en BFOA han optado por utilizar suma ponderada de objetivos para resolver el problema como un mono-objetivo [82, 83, 84].

- Las propuestas basadas en BFOA para resolver problemas de optimización numérica usan principalmente técnicas de primera generación para el manejo de restricciones. La técnica más popular es función de penalización [79, 85, 80, 86, 87, 88], seguida de la ténica de penalización de muerte [81, 84] y operadores especiales [82, 83]. Y por último, las reglas de factibilidad

Autores	Técnica de manejo de restricción	Manejo de tamaño de paso	Caracsterística especial
Primera generación			
Praveena et al. [79]	Función de penalización	Aleatorio	Hibrido BFOA-PSO-DE
Chunguang et al. [80]	Función de penalización	Definido por el usuario	Ninguna
Wei et al. [81]	Penalización de muerte	Definido por el usuario	Asistente sustituto
Pangrahi et al. [82]	Operador especial	Definido por el usuario	Operador de mejora del giro-nado
Pangrahi et al. [83]	Operador especial	Definido por el usuario	Operador de mejora del giro-nado
Deshpande et al. [84]	Penalización de muerte	Definido por el usuario	Espacio de búsqueda entero
Pandit et al. [85]	Función de penalización	Dinámico	Población pequeña
Vaisakh et al. [86]	Función de penalización	Aleatorio	Hibrido BFOA-PSO-DE
Saber [87]	Función de penalización	Adaptativo usando Información de factibilidad	Hibrido PSO-BFOA
Niu et al. [88]	Función de penalización	Dinámico	Ninguna
Segunda generación			
Mezura-Montes and Hernández-Ocaña [78]	Reglas de factibilidad	Definido por fórmula y fijado durante la ejecución	Simplificación del BFOA
Mezura-Montes et al. [89]	Reglas de factibilidad	Definido por fórmula y fijado durante la ejecución	Simplificación del BFOA original para optimización multi-objectivo
Mezura-Montes and López-Dávila [34]	Reglas de factibilidad	Adaptativo	Búsqueda local

Tabla 2.1: Características principales de cada propuesta basada en BFOA que utiliza alguna técnica de manejo de restricción.

una de las técnicas de segunda generación para el manejo de restricciones es usada en BFOA [78, 89, 34].

- El parámetro más estudiado y modificado en las propuestas basadas en BFOA es el tamaño de paso. En la mayoría de las propuestas el tamaño de paso es definido por el usuario [82, 80, 81, 83, 84, 43, 46, 18, 21, 18, 48, 50, 54, 55, 56, 63, 65, 68, 69, 71, 75, 81, 80, 74]. En otras propuestas se han diseñado expresiones para asignar el valor al tamaño de paso basado en las características del problema [78, 89, 54, 45, 17, 53]. Otras propuestas generan el valor del tamaño de paso de manera aleatoria [79, 86], ajustes dinámicos [14, 51, 62, 61, 85, 88] o adaptativos [34, 87, 29, 15, 51, 26, 73, 62, 64, 91, 63, 49, 57, 76, 89, 13, 45, 17, 49, 90, 57, 59, 60, 64, 66, 73, 36, 83]. Cabe mencionar que Saber [87] propone una mecanismo adaptativo, el cual usa información factible para calcular el valor del tamaño de paso. Aunque el tamaño de paso sea calculado de manera dinámica, adaptativa o aleatoria siempre hay parámetros que el usuario tiene que definir.

- El operador de nado-giro también ha sido modificado en algunas propuestas, por ejemplo, una velocidad en vez de una búsqueda de dirección aleatoria [79, 86], mecanismo de guía [82, 83, 32, 75] y operador de agrupamiento [78, 89, 34, 55].

- El proceso de reproducción, también se ha sido modificado: se disminuye el número de veces en que se hace presente este proceso o en definitiva lo eliminan del algoritmo, ya que para algunos tipos de problemas este proceso hace que el algoritmo converja demasiado rápido, haciendo caer en óptimos locales [82, 85, 89, 81, 83, 34, 87, 15, 67]. Por otra parte, el proceso de reproducción es aumentado gradualmente [27] para una convergencia más rápida.

- El proceso de eliminación-dispersión ha sido poco modificado, sin embargo, éste es modificado usando el operador principal de PSO [72], eliminando [53] o reducida su presencia [36, 34].

- Pequeñas poblaciones en BFOA son presentes [85] como también versiones de espacios de búsqueda discreta [84, 15].

- BFOA ha sido reducido en ciclos *for* y parámetros definidos por el usuario [34, 59, 36, 89].
- BFOA ha sido combinado con otros algoritmos, por ejemplo PSO-ED [79, 86], PSO [87, 31, 32], ED [45] y Hooke-Jeeves como un operador de búsqueda local [34]. También BFOA es combinado con lógica difusa-sistema inmune [44], AG [30].

BFOA también ha sido aplicado para resolver problemas combinatorios como la generación de horarios [90] y descrubrimiento o extracción de secuencias de ADN (motif) [91], [92].

Capítulo 3

Algoritmo Modificado basado en el forrajeo de bacterias (MBFOA)

En este capítulo se describen cada uno de los procesos del algoritmo Modificado basado en el forrajeo de bacterias.

3.1. Mecanismo de MBFOA

MBFOA es un algoritmo propuesto en [36] derivado de BFOA [11] en el cual se implementan varios mecanismos que permiten al algoritmo resolver problemas de optimización mono-objetivo con restricciones.[36].

Los mecanismo del MBFOA son los siguientes:

1. Un sólo ciclo para incluir el ciclo quimiotáxico, la reproducción y eliminación-dispersión.

El ciclo de generaciones es controlado por el número máximo de generaciones (GMAX) en el cual las bacterias realizan su ciclo quimiotáxico. Después de esto se lleva a cabo una sola reproducción y una eliminación-dispersión dentro de un mismo ciclo de generación. En el paso de eliminación-dispersión sólo se elimina a la peor bacteria de la población.

2. Tamaño de paso diferente para cada variable de diseño.

El tamaño de paso $C(i)$ es definido considerando los límites inferior y superior L_i y U_i para cada variable de decisión x_i, usando la ecuación 3.1

$$C(i) = R * \left(\Delta\vec{x}_i/\sqrt{n}\right) \tag{3.1}$$

Donde C(i) es el tamaño de paso que, en este caso, ya no está definido por el usuario, $\Delta\vec{x}_i$ es la diferencia $U_i - L_i$, n es el número de variables en el problema de optimización y R es el porcentaje del total del tamaño de paso a ser usado, un tamaño de paso inicial pequeño suele ser conveniente en optimización restringida, conclusiones que se reportan en [36].

En este mismo proceso se asegura que los elementos en $\vec{x}$ (variables de diseño) no rebasen sus límites establecidos $L_i \leq x_i \leq U_i, \forall i = 1, ..., n$, mediante el siguiente módulo:

Si x_i<Li entonces:

$$x_i = 2L_i - x_i \tag{3.2}$$

Si x_i>Ui entonces:

$$x_i = 2U_i - x_i \tag{3.3}$$

3. El manejo de las restricciones del problema es usando las reglas de factibilidad de Deb [37], estas reglas son usadas en el proceso quimiotáxico (específicamente en el nado), reproducción y eliminación-dispersión.

4. Comunicación entre las bacterias y la mejor bacteria de la población de una generación.

En ciertas iteraciones del ciclo quimiotáxico (a la mitad y al final) se permite que cada bacteria de la población se oriente y nade orientada por la mejor bacteria de la población (efecto de swarming) calculado como se indica en la ecuación 3.4:

$$\theta^i(j+1, G) = \theta^i(j, G) + \beta(\theta^B(G) - \theta^i(j, G)) \tag{3.4}$$

Donde $\theta^i(j+1, G)$ es la nueva posición de la bacteria i, $\theta^B(G)$ es la actual posición de la mejor bacteria en la generación G hasta ahora y β es un parámetro llamado factor de escalamiento, el cual regula qué tan cerca estará la bacteria i de la mejor bacteria θ^B.

Los pasos restantes del ciclo quimiotáxico son realizados como se indica en la ecuación 3.5:

$$\theta^i(j+1, G) = \theta^i(j, G) + C_(i)\phi(i) \tag{3.5}$$

donde la dirección aleatoria de las bacterias es modelada por la ecuación 3.6:

$$\phi(i) = \frac{\Delta(i)}{\sqrt{\Delta(i)^T \Delta(i)}} \tag{3.6}$$

$\Delta(i)^T$ es un vector aleatorio generado con elementos dentro de un intervalo:$[-1, 1]$. Los parámetros de MBFOA se describen en la Tabla 3.1.

Nombre	**Símbolo**	**Descripción**
Bacterias	S_b	Número de soluciones en la población (bacterias).
Generaciones	GMAX	Número de generaciones en el que se ejecutará el procedimiento quimiotáxico, de reproducción y eliminación-dispersión.
Pasos quimiotáxico	N_c	Número de veces que cada bacteria en la población podrá nadar o girar.
Reproducción	S_r	Número de bacterias a reproducirse.
Escalamiento	β	Factor de escalamiento que determina cuánto avanzará una bacteria hacia la mejor bacteria de la población.
Tamaño de paso	R	Porcentaje del tamaño de paso que una bacteria avanzará durante su ciclo quimiotáxico.

Tabla 3.1: Descripción de los parámetros del MBFOA

El pseudocódigo del MBFOA es presentado en el algoritmo 1, en la cual se detallan los mecanismos antes mencionados. Los parámetros de entrada son el número de bacterias S_b, límite del paso quimiotáxico N_c, número de bacterias a reproducirse S_r, factor de escalamiento β, porcentaje del tamaño de paso R y el número de generaciones GMAX.

```
Crear una población inicial de bacterias θ^i(j,0) ∀i, i = 1,...,S_b
Evaluar f(θ^i(j,0)) ∀i, i = 1,...,S_b
for G=1 to GMAX do
    for i=1 to S_b do
        for j=1 to N_c do
            Realizar el paso quimiotáxico (giro-nado o giro-giro) para la bacteria θ^i(j,G) usando las reglas de
            factibilidad y las Ec.3.5 y 3.4
        end
    end
    Realizar el paso de reproducción para la eliminación de S_r (mitad) peores bacterias y duplicar la otra mitad,
    basado en las reglas de factibilidad.
    Eliminar a la peor bacteria θ^w(j,G) de la población, basado en las reglas de factibilidad y generar una nueva
    aleatoriamente.
end
```

Algoritmo 1: Pseudocódigo de MBFOA.

Capítulo 4

MBFOA y técnicas de manejo de restricciones

En este capítulo se reportan las adaptaciones realizadas al algoritmo MBFOA para conocer qué técnica de manejo de restricciones permite mejorar su rendimiento en el proceso de optimización. Las propuestas derivadas son probadas en problemas de pruebas para observar su rendimiento.

4.1. Técnicas de manejo de restricciones

En la literatura especializada se encuentran distintas técnicas para el manejo de restricciones [93, 94], donde los algoritmos que más han usado tales técnicas son los AEs, sin embargo recientemente los ICs también las están usando para poder resolver problemas con restricciones y mejorar la calidad de sus resultados. Basado en una revisión actualizada del estado del arte en optimización numérica con restricciones inspirada en la naturaleza [40] se presenta un listado de las técnicas de manejo de restricciones divididas en dos generaciones.

Primera generación:

- **Función de penalización** [41]: basada en la disminución del valor de aptitud de soluciones no factibles dependiendo de la violación de restricciones para favorecer a las soluciones factibles en el proceso de selección. Las restricciones son unidas a la función objetivo para generar una función extendida. Además se utilizan factores de penalización donde el usuario define el valor de estos factores que determinan la severidad del castigo a una solución por violar restricciones.

- **Decodificadores** [95]: basados en la transformación de la región factible original del problema en un espacio más adecuado para llevar a cabo la búsqueda.

- **Operadores especiales** [96]: diseñados para problemas particulares con el objetivo de mantener la factibilidad de las nuevas soluciones generadas a partir de soluciones factibles.

- **Separación de la función objetivo y las restricciones** [97]: A diferencia de la técnica de función de penalización, donde los valores de la función objetivo y las restricciones son combinados en un solo valor, este enfoque usa estos valores por separado como criterio de selección dependiendo de la situación en la búsqueda.

Estas técnicas presentan ciertas desventajas por ejemplo, un sesgo inadecuado al separar la función objetivo de las restricciones, ajuste cuidadoso de parámetros sensibles en el caso de la función de penalización, dificultad para generar operadores especiales, e incluso alto costo computacional e implementaciones difíciles en el caso de los decodificadores.

Segunda generación:

- **Reglas de factibilidad** [37]: basadas en un conjunto de tres reglas donde la función objetivo y la suma de violación de restricciones son manejadas por separado sin la adición de parámetros definidos por el usuario. Las reglas son:
  1. Entre dos soluciones factibles, aquella con el mejor valor de la función objetivo es seleccionada.
  2. Entre una solución factible y otra no factible, la factible es seleccionada.
  3. Entre dos soluciones no factibles, aquella con la menor suma de violación de restricciones es seleccionada.
- **Jerarquización estocástica** [98]: un mecanismo de ordenamiento basado en factibilidad; sin embargo, basado en un parámetro definido por el usuario, algunas soluciones son ordenadas basadas solo por el valor de la función objetivo independientemente de la factibilidad.
- **Método ϵ-constrained** [99]: transforma un problema de optimización numérica con restricciones en un problema de optimización numérica sin restricciones usando una tolerancia llamada ϵ para considerar como factible soluciones ligeramente no factibles.
- **Nuevas funciones de penalización** [100]: funciones de penalización adaptativas o dinámicas que no requieren de factores de penalización definidos por el usuario.
- **Nuevos operadores especiales** [101]: a diferencia de los operadores de primera generación, éstos se enfocan en el muestreo de las regiones de interés en el espacio de búsqueda, por ejemplo, los límites de la región factible.
- **Conceptos multi-objetivo** [102]: transforma un problema de optimización numérica con restricciones en un problema de optimización multi-objetivo sin restricciones donde la dominancia de Pareto o jerarquización de Pareto son usadas como criterio de selección.
- **Ensamble de técnicas de manejo de restricciones** [103]: una combinación de dos o más técnicas dentro de un algoritmo inspirado en la naturaleza.

Estas diferentes técnicas fueron y son usadas por los algoritmos inspirados en la naturaleza, cabe mencionar que los primeros en usar éstas técnicas fueron el Algoritmo Genético (AG) [85], Evolución Diferencial (ED) [83] y Optimización mediante el cúmulo de partículas (PSO por sus siglas en Ingles) [66]. A continuación en el estado del arte de BFOA se encuentran algunas versiones que hacen uso de estas técnicas de manejo de restricciones.

4.2. MBFOA con cuatro técnicas para el manejo de restricciones

Las técnicas de manejo de restricciones, seleccionadas del estado del arte, para ser probadas en MBFOA son aquellas que han sido usadas en otras propuestas basadas en BFOA, como se resume en la Tabla 2.1.3, aunque sólo se prueban las técnicas donde su implementación es clara y viable de implementar, las consideradas son:

- Función de penalización (FP) [80]
- Penalización de muerte (PM)[81] (derivada de función de penalización) consiste en penalizar con un valor muy alto a las soluciones que violan restricciones o eliminarlas (y generar una nueva solución aleatoria hasta que esta sea factible) dando prioridad a las soluciones factibles de mantenerse en evaluaciones siguientes.
- Penalización adaptativa (PA) [104] que consiste en calcular el valor de penalización para cada solución de acuerdo al promedio de las restricciones violadas y soluciones factibles de la población en ejecución.
- Reglas de factibilidad (RF) [37]
- ϵ-Constraint (ϵc)[99]

MBFOA es una algoritmo basado en BFOA que usa las Reglas de Factibilidad de Deb [37] pero no ha sido probado con ninguna otra técnica de manejo de restricciones. En la siguiente serie de experimentos se prueba MBFOA usando Función de penalización, Penalización de muerte, ϵ-Constraint (ϵ-c), Penalización adaptativa y las Reglas de factibilidad como mecanismos de manejo de restricciones.

A MBFOA fue incorporado un mecanismo de tolerancia dinámica inspirado en [105] para que tenga un mejor desempeño en funciones con presencia de restricciones de igualdad y desigualdad. El mecanismo es utilizado al inicio de cada generación del algoritmo usando la siguiente ecuación 4.1:

$$\xi(G+1) = \frac{\xi(G)}{dec} \tag{4.1}$$

donde G es la generación actual del algoritmo y dec es la tasa de valor decreciente por cada generación ($dec > 1$). Esto permite que el algoritmo tenga una mayor región factible que la original y por consecuencia las restricciones son satisfechas fácilmente en las primeras generaciones del algoritmo hasta que $\xi = \gamma$, donde $\gamma = 0{,}0001$ es el valor permitido para violación de restricciones de igualdad según lo establecido en la literatura especializada [106].

En el caso de la técnica de manejo de restricciones ϵc, la tolerancia llamada ϵ, para considerar como factible soluciones ligeramente no factibles, es inicializada con el valor de la suma de violación de restricciones de la peor bacteria de la población, es decir, con la suma de violación máxima que existe en la población ϵMax en la primera generación del algoritmo. ϵ disminuye gradualmente durante las generaciones del algoritmo hasta que el número de evaluaciones ($evals$) es 150,000 usando la ecuación 4.2 basada en la propuesta de control de nivel de ϵ encontrada en [99]. Después de este límite de evaluaciones, ϵ toma el valor 0.

$$\epsilon = \epsilon Max(1 - \frac{evals}{150,000})^5; \tag{4.2}$$

La presencia del proceso de reproducción es disminuida para evitar una convergencia prematura. Este proceso se hace presente después de un determinado ciclo de generaciones definido por el usuario en el parámetro (RepCycle).

Para probar las distintas técnicas de manejo de restricciones en MBFOA, los parámetros usados se presentan en la Tabla 4.1. Las propuestas fueron probadas en el conjunto de 24 funciones de prueba del CEC2006, un resumen de las funciones es presentado en los Anexos 8 en la Tabla 8.1.

Parámetro	**valor**
Población	41
Ciclo quimiotáxico	20
Tamaño de paso	0.0066470374
Factor de escalamiento	1.433092
Reproducción	8
Factor de penalización	0.2360226
ξ	1.6
dec	1.09

Tabla 4.1: Parámetros de MBFOA usados con las diferentes técnicas de manejo de restricciones.

En la Tablas 4.2 y 4.3 se encuentran las estadísticas básicas de cada propuesta de MBFOA usando distinta técnica de manejo de restricciones y se incluye el número de ejecuciones factibles de cada propuesta de las 30 ejecuciones independientes realizadas cada una con 239, 773 evaluaciones.

Con base en los resultados y estadísticas se observa:

Funciones de prueba que se resuelven al satisfacer la condición $f(\overrightarrow{x}) - f(\overrightarrow{x^*}) \leq 0{,}0001$

- MBFOA-RF encuentra resultados similares al óptimo global conocido en las funciones de prueba G3, G8, G11, G12 y G24.
- MBFOA-FP encuentra resultados similares al óptimo global conocido en las funciones de prueba G8 y G12.
- MBFOA-PM encuentra solo en la función de prueba G12 resultados similares al óptimo global conocido.
- MBFOA-PA y MBFOA-ϵc no resuelven ninguna función de prueba.

Calidad de los resultados (Con base en la media y al mejor valor encontrado)

- Para la función G8 la propuesta MBFOA-RF tiene una mejor calidad de resultados comparada con MBFOA-FP.
- Para función G12 la propuesta MBFOA-RF tiene una mejor calidad de resultados que MBFOA-PM y éste último es mejor que MBFOA-FP.

Funciones de prueba donde se encuentran soluciones factibles

- MBFOA-RF obtiene soluciones factibles a 17 de las 24 funciones de prueba.
- MBFOA-PA obtiene soluciones factibles a 14 de las 24 funciones de prueba
- MBFOA-ϵc obtiene soluciones factibles a 6 de las 24 funciones de prueba
- MBFOA-PM obtiene soluciones factibles a 5 de las 24 funciones de prueba
- MBFOA-FP obtiene soluciones factibles a 4 de las 24 funciones de prueba

Prob./ Óptimo	Estadísticas	MBFOA-FP	MBFOA-PM	MBFOA-PA	MBFOA-RF	MBFOA-cc
	Mejor	-	-	-6.029699	**-9.885358**	-
G1	Media	-	-	-2.826448	**-4.913382**	-
-15	Desv.Est.	-	-	**1.841382**	2.108167	-
	Peor	-	-	1.007911	0.217578	-
	E.Factibles	-	-	18	30	
	Mejor	-0.3846964	-0.411424	-0.190098	**-0.449230**	-0.429108
G2	Media	-0.265854	-0.318927	-0.150169	**-0.332766**	-0.322699
-0.80361910412559	Desv.Est.	0.044089	0.041137	**0.016636**	0.047011	0.042339
	Peor	-0.198314	-0.251074	-0.123874	-0.266820	-0.267987
	E.Factibles	30	30	30	30	30
	Mejor	-	-	-0.043296	**-1.000472**	-
G3	Media	-	-	-0.006146	**-1.000440**	-
-1.00050010001000	Desv.Est.	-	-	**0.011700**	2.302E-5	-
	Peor	-	-	-9.176E-7	-1.000392	-
	E.Factibles	-	-	22	30	
	Mejor	-	-	-30176.641783	**-30665.050709**	-30603.594158
G4	Media	-	-	-28862.954153	**-30663.331315**	-30364.450064
-30665.53867178332	Desv.Est.	-	-	629.859015	**4.186242**	199.896680
	Peor	-	-	-27684.648511	-30641.321606	-30036.829744
	E.Factibles	-	-	30	30	10
	Mejor	-	-	-	-	-
G5	Media	-	-	-	-	-
5126.4967140071	Desv.Est.	-	-	-	-	-
	Peor	-	-	-	-	-
	E.Factibles	-	-	-	-	-
	Mejor	-	-6934.613548	-5876.830972	**-6960.833737**	-
G6	Media	-	-6768.759374	-4293.010801	**-6950.804575**	-
-6961.81387558015	Desv.Est.	-	101.338114	1183.506619	**12.122236**	-
	Peor	-	-6568.919449	-3080.031844	-6902.059482	-
	E.Factibles	-	14	4	30	
	Mejor	-	-	-	**24.726052**	-
G7	Media	-	-	-	**25.910263**	-
24.30620906818	Desv.Est.	-	-	-	**0.835754**	-
	Peor	-	-	-	27.821808	-
	E.Factibles	-	-	-	30	-
	Mejor	-0.095628	-	-0.056450	**-0.095825**	-
G8	Media	-0.048499	-	-0.004605	**-0.095825**	-
-0.095825	Desv.Est.	**0.032205**	-	0.035367	1.603E-12	-
	Peor	-0.025395	-	0.101396	-0.095825	-
	E.Factibles	10	-	30	30	
	Mejor	-	680.679327	1091.893954	**680.653436**	-
G9	Media	-	680.909138	921214.938277	**680.706034**	-
680.630057	Desv.Est.	-	0.482951	1427591.544936	**0.042509**	-
	Peor	-	682.183394	4674959.504448	680.841363	-
	E.Factibles	-	9	26	30	-
	Mejor	-	-	15463.642803	**7082.964009**	-
G10	Media	-	-	18737.618033	**7356.790777**	-
7049.24802052867	Desv.Est.	-	-	3013.022309	**488.860613**	-
	Peor	-	-	21393.977063	9257.094046	-
	E.Factibles	-	-	3	30	-
	Mejor	-	-	0.751507	**0.749900**	-
G11	Media	-	-	0.805752	**0.749901**	-
0.7499	Desv.Est.	-	-	0.063379	**2.115E-6**	-
	Peor	-	-	0.998039	0.749908	-
	E.Factibles	-	-	30	30	-
	Mejor	-0.999999	-0.999999	-0.993038	**-0.999999**	-0.812171
G12	Media	-0.962065	-0.998871	-0.911926	**-0.999247**	-0.812171
-1	Desv.Est.	0.035309	0.002300	0.069291	**0.001947**	-
	Peor	-0.905914	-0.994360	-0.673077	-0.994359	-0.812171
	E.Factibles	13	25	30	30	1

Tabla 4.2: Parte 1 de las estadísticas básicas de los resultados de MBFOA usando distinta técnica de manejo de restricciones.

Prob./ Óptimo	Estadísticas	MBFOA-FP	MBFOA-PM	MBFOA-PA	MBFOA-RF	MBFOA-cc
	Mejor	-	-	-	-	-
G13	Media	-	-	-	-	-
0.053941	Desv.Est.	-	-	-	-	-
	Peor	-	-	-	-	-
	E.Factibles	-	-	-	-	-
	Mejor	-	-	-	**-42.534548**	-
G14	Media	-	-	-	**-38.684487**	-
-47.764888	Desv.Est.	-	-	-	**2.519629**	-
	Peor	-	-	-	-35.097562	-
	E.Factibles	-	-	-	10	-
	Mejor	-	-	967.331435	**961.715343**	961.738753
G15	Media	-	-	979.565842	**961.717716**	961.738753
961.715022	Desv.Est.	-	-	6.218860	**0.001571**	-
	Peor	-	-	997.882502	961.720933	961.738753
	E.Factibles	-	-	30	30	1
	Mejor	-1.902837	-	-1.369757	**-1.903357**	-
G16	Media	-1.718565	-	-1.170904	**-1.887545**	-
-1.905155	Desv.Est.	0.125968	-	0.241890	**0.056356**	-
	Peor	-1.498474	-	-0.822953	-1.612253	-
	E.Factibles	18		5	30	-
	Mejor	-	-	-	-	-
G17	Media	-	-	-	-	-
8853.539674	Desv.Est.	-	-	-	-	-
	Peor	-	-	-	-	-
	E.Factibles	-	-	-	-	-
	Mejor	-	-	-	**-0.859667**	-
G18	Media	-	-	-	**-0.730242**	-
-0.866025	Desv.Est.	-	-	-	**0.118340**	-
	Peor	-	-	-	-0.492514	-
	E.Factibles	-	-	-	27	-
	Mejor	-	54.915929	711.98543	**49.473018**	366.271279
G19	Media	-	89.700988	6552.661664	**117.292903**	659.871468
32.655592	Desv.Est.	-	**33.611884**	3160.587357	73.266606	233.839430
	Peor	-	191.877301	12247.943085	320.696077	1122.159935
	E.Factibles	-	30	30	30	10
	Mejor	-	-	-	-	-
G20	Media	-	-	-	-	-
0.204979400285636	Desv.Est.	-	-	-	-	-
	Peor	-	-	-	-	-
	E.Factibles	-	-	-	-	-
	Mejor	-	-	-	-	-
G21	Media	-	-	-	-	-
193.724510070035	Desv.Est.	-	-	-	-	-
	Peor	-	-	-	-	-
	E.Factibles	-	-	-	-	-
	Mejor	-	-	-	-	-
G22	Media	-	-	-	-	-
236.430975504001	Desv.Est.	-	-	-	-	-
	Peor	-	-	-	-	-
	E.Factibles	-	-	-	-	-
	Mejor	-	-	-	-	-
G23	Media	-	-	-	-	-
-400.055099999999584	Desv.Est.	-	-	-	-	-
	Peor	-	-	-	-	-
	E.Factibles	-	-	-	-	-
	Mejor	-	-	-5.449168	**-5.508006**	-5.505420
G24	Media	-	-	-4.7841050205448825	**-5.507687062457061 5**	-5.175550
-5.50801327159536	Desv.Est.	-	-	0.477710	**2.831E-4**	0.644373
	Peor	-	-	-3.728328	-5.506601	-4.031639
	E.Factibles	-	-	30	30	9

Tabla 4.3: Parte 2 de las estadísticas básicas de los resultados de MBFOA usando distinta técnica de manejo de restricciones.

En general:

- La propuesta de MBFOA usando Reglas de factibilidad es la que obtiene mejores resultados en comparación de las demás técnicas.
- La Función de penalización adaptativa es la segunda mejor técnica que mejora el rendimiento del algoritmo.
- Los resultados obtenidos no son competitivos en comparación al mejor conocido encontrado en [106].

Capítulo 5

MBFOA y distintos tamaño de paso

En este capítulo se describen cuatro mecanismos para el calculo del tamaño de paso, los cuales son: dinámico, adaptativo, estático y dinámico. Cada de una de las propuestas derivadas es probada en problemas de prueba para observar su rendimiento.

5.1. MBFOA usando un tamaño de paso dinámico

MBFOA presenta 6 parámetros que deben ser definidos por el usuario. Uno de los parámetros más sensibles es el tamaño de paso de acuerdo a las distintas conclusiones encontradas en la literatura especializada [39]. A diferencia de MBFOA, en esta propuesta el tamaño de paso inicial $C(i,G)$ es generado aleatoriamente con la ecuación de direcciones de giro 3.6 entre un rango de valores de [-1,1]. El tamaño de paso disminuirá generacionalmente y por lo tanto, el parámetro de porcentaje del total del tamaño de paso R ya no será usado.

Seis propuestas para controlar la disminución del tamaño de paso generacional son propuestas cada una son funciones lineales y no-lineales que a continuación se presentan [107]:

- Propuesta 1:

$$C(i,G+1) = C(i,G)_k * \frac{G}{GMAX}^4, k = 1, ..., n \tag{5.1}$$

- Propuesta 2:

$$C(i,G+1) = C(i,G)_k * \frac{G}{GMAX}^2, k = 1, ..., n \tag{5.2}$$

- Propuesta 3:

$$C(i,G+1) = C(i,G)_k * \frac{G}{GMAX} - \frac{sin(2\pi(G - GMAX))}{6{,}3}, k = 1, ..., n \tag{5.3}$$

- Propuesta 4:

$$C(i,G+1) = C(i,G)_k * \frac{G}{GMAX}, k = 1, ..., n \tag{5.4}$$

- Propuesta 5:

$$C(i,G+1) = C(i,G)_k * \frac{G}{GMAX}^{1/2}, k = 1,...,n \tag{5.5}$$

- Propuesta 6:

$$C(i,G+1) = C(i,G)_k * \frac{G}{GMAX}^{1/4}, k = 1,...,n \tag{5.6}$$

donde $C(i,G+1)$ es el nuevo vector de tamaño de paso, G es la generación actual del algoritmo y $GMAX$ es el número máximo de generaciones permitidas al algoritmo. Cada una de las funciones lineales y no-lineales hacen que el tamaño de paso disminuya de diferente manera, por ejemplo, la propuesta 6 se aproxima al valor 0 de forma más rápida pero sin aproximarse tanto como la propuesta 5. La propuesta 4 es completamente lineal y hace que el valor del tamaño de paso disminuya constantemente pero su aproximación al valor 0 es cada vez más en comparación a las propuestas 5 y 6. Las propuestas 1, 2 y 3 disminuyen el tamaño de paso hasta llegar a valores muy cercanos al valor 0 (mucho más que las propuestas 4, 5 y 6). Las propuestas 1, 2 y 3 hacen que el tamaño de paso en un gran número de generaciones tenga un valor muy pequeño. Algunas conclusiones de autores encontradas en el estado del arte [39] a firman que el tamaño de paso con valor muy cercano a 0 permite un mejor rendimiento del algoritmo, esto es debido a que las bacterias con nados muy pequeños son capaces de explorar y explotar una región del espacio de búsqueda.
El tamaño de paso dinámico fue probado en MBFOA con los mecanismos de manejo de restricciones que obtuvieron mejores resultados, las cuales son MBFOA-RF y MBFOA-PA, con tamaño de paso dinámico. Ambas propuestas son sometidas a las 24 funciones de prueba 8.1 con los parámetros presentados en la Tabla 5.1.

Parámetros	**valores**
Población	49
Ciclo quimiotáxico	19
Factor de escalamiento	1.3256403
Reproducción	2
ξ	1.6
dec	1.09

Tabla 5.1: Parámetros usados con las diferentes propuestas usando tamaño de paso dinámico.

La mejor solución encontrado por cada una de las 6 propuestas usando RF y PA se presenta en las Tablas 5.2 y 5.3. En las Tablas 5.4 y 5.5 se presenta el valor de la media de las 30 ejecuciones independientes realizadas cada una con 239, 573 evaluaciones.

Con base en los resultados obtenidos en este experimento se puede deducir que la propuesta MBFOA-RF con tamaño de paso $(\frac{G}{GMAX})^4$ es quien obtiene mejores resultados en comparación con los resultados de las demás propuestas, y a partir de esta sección se identificará con el nombre DyMBFOA.

El tamaño de paso dinámico en DyMBFOA probado en CEC2006

DyMBFOA es nuevamente probado, ahora en los dos conjuntos de funciones de prueba CEC2006 y CEC2010, y comparado con otros algoritmos encontrados en el estado del arte que resuelven estas mismas funciones de prueba. Los parámetros utilizados se presentan en la Tabla 5.6.

MBFOA-RF					
$(\frac{G}{GMAX})^4$	$(\frac{G}{GMAX})^2$	$(\frac{G}{GMAX}) - \frac{sin(2\pi(G-GMAX))}{6.3}$	$(\frac{G}{GMAX})$	$(\frac{G}{GMAX})^{1/2}$	$(\frac{G}{GMAX})^{1/4}$
-14.661747	-13.287895	-14.049993	-13.306446	-13.496674	-12.999869
-0.736833	-0.738724	-0.684338	**-0.773271**	-0.696022	-0.594892
-0.998089	-0.989389	-0.803435	-0.953746	-0.873169	-0.680238
-30665.538663	-30665.532031	-30665.536160	-30664.977517	-30662.681574	-30645.387208
5126.541282	-	5143.319364	-	-	-
-6961.813717	-6961.726303	-6961.762944	-6960.541935	-6949.935535	-6955.353302
24.322076	24.332632	24.389831	24.806644	26.315269	36.348766
-0.095825	-0.095825	**-0.095825**	-0.095825	-0.095825	-0.095825
680.635389	680.637046	**680.631817**	680.769102	682.287197	686.476841
7055.742970	7079.017566	7086.101778	7206.780483	7507.093124	8185.471933
0.749900	0.749900	0.749900	0.749900	0.749904	0.749903
-1	-0.999999	**-1**	-0.999999	-0.999999	-0.999999
0.054272	0.058596	0.056036	0.598457	0.553014	0.554011
-46.362633	**-46.571999**	-46.550139	-41.939052	0	0
961.715172	961.715199	961.715198	961.717479	961.740919	961.726962
-1.905091	-1.904946	-1.904709	-1.902735	-1.899853	-1.889163
8878.957711	-	-	-	-	-
-0.866019	-0.865302	-0.865814	-0.847621	-0.725456	-0.566260
38.208404	**35.078713**	36.287918	37.895713	43.833090	52.947237
-	-	-	-	-	-
-	-	-	-	-	-
-	-	-	-	-	-
-	-	-	-	-	-
-5.508013	-5.508012	-5.508013	-5.507977	-5.507772	-5.507787

Tabla 5.2: Mejor valor encontrado por la propuesta MBFOA-RF usando distinto tamaño de paso dinámico.

MBFOA-PA					
$(\frac{G}{GMAX})^4$	$(\frac{G}{GMAX})^2$	$(\frac{G}{GMAX}) - \frac{sin(2\pi(G-GMAX))}{6.3}$	$(\frac{G}{GMAX})$	$(\frac{G}{GMAX})^{1/2}$	$(\frac{G}{GMAX})^{1/4}$
-6.064558	-4.956239	-6.294801	**-8.499464**	-6.518973	-5.919339
-0.185629	**-0.219593**	-0.205542	-0.217790	-0.176807	-0.172261
-0.189965	**-0.314134**	-0.114979	-0.094810	-0.2143431	-0.133672
-29529.788504	-29958.298504	-29766.092164	-29773.593688	**-29975.607180**	-29709.908755
-	-	-	-	-	-
-5862.397135	-6339.914828	-5825.386065	**-6561.515474**	-6287.883003	-5785.294557
442.074579	381.505642	448.086899	349.264831	**274.273804**	338.832316
-0.081058	-0.072436	-0.053864	-0.079345	**-0.088045**	-0.087994
1555.106037	1249.903247	1264.169339	1133.087292	1243.325346	**1131.869905**
14305.531303	**11106.700605**	14903.379214	12059.815499	12134.882790	13217.668247
0.756304	0.755255	0.750102	0.750165	**0.750028**	0.750108
-0.987787	-0.987828	-0.984194	**-0.991229**	-0.990812	-0.950101
-	-	-	-	-	-
-	-	-	-	-	-
963.199123	973.549017	972.697357	972.660065	965.894270	965.660556
-1.733326	-1.715362	-1.490172	-1.586219	-1.678117	-1.584860
-	-	-	-	-	-
-0.472937	-0.395386	-0.453433	-0.463323	-0.297360	-0.383137
713.424613	1180.344289	2115.828417	1674.163340	1296.993642	1025.331651
-	-	-	-	-	-
-	-	-	-	-	-
-	-	-	-	-	-
-	-	-	-	-	-
-4.392277	-4.319755	-4.530907	-4.851708	**-5.107503**	-4.805797

Tabla 5.3: Mejor valor encontrado por la propuesta MBFOA-PA usando distinto tamaño de paso dinámico.

En las Tablas 5.7 y 5.8 se presentan los resultados obtenidos por la propuesta DyMBFOA en el CEC2006 y se compara con los resultados de otros algoritmos encontrados en el estado del arte. Todos los algoritmos comparados permiten una tolerancia para las restricciones de igualdad de 1E-04. Sin embargo, el número de evaluaciones usadas de APF-GA y MDE fueron 500,000 mientras que Memetic-SAMSDE y DyMBFOA fueron 240,000. Otra diferencia es el número de ejecuciones independientes realizadas, todos los algoritmos realizaron 25 ejecuciones excepto Memetic-SAMSDE que realizó 30.
Con base en los resultados obtenidos por la propuesta se obtiene el siguiente análisis: **Funciones de prueba que resuelve al satisfacer la condición** $f(\overrightarrow{x}) - f(\overrightarrow{x^*}) \leq 0{,}0001$:

- Memetic-SAMSDE resuelve casi todas las funciones de prueba excepto g19 y g20.

MBFOA-RF					
$(\frac{G}{GMAX})^4$	$(\frac{G}{GMAX})^2$	$(\frac{G}{GMAX}) - \frac{sin(2\pi(G-GMAX))}{6,3}$	$(\frac{G}{GMAX})$	$(\frac{G}{GMAX})^{1/2}$	$(\frac{G}{GMAX})^{1/4}$
-9.350	-9.105	**-10.272**	-9.698	-9.963	-8.631
-0.476	**-0.516**	-0.458	-0.508	-0.458	-0.444
-0.991	-0.896	-0.641	-0.748	-0.624	-0.478
-30665.537	-30665.474	-30665.495	-30662.952	-30648.515	-30615.501
5244.787	-	**5143.319**	-	-	-
-6961.812	-6960.653	-6961.310	-6859.264	-6640.084	-6118.825
24.421	24.518	24.558	25.299	29.895	43.349
-0.095	-0.095	-0.095	-0.095	-0.095	-0.095
680.664	680.688	680.702	681.079	685.037	698.375
7144.741	7164.168	7221.559	7373.602	8122.276	9031.228
0.749	0.749	0.749	0.750	0.753	0.751
-1	-0.999	-0.999	-0.999	-0.999	-0.999
0.171	0.317	0.277	0.746	0.847	0.845
-44.142	-43.664	-43.980	-41.939	0	0
961.715	961.715	961.715	961.726	961.810	962.000
-1.903	-1.902	-1.902	-1.897	-1.872	-1.845
8983.302	-	-	-	-	-
-0.865	-0.863	-0.862	-0.824	-0.594	-0.292
60.135	**54.657**	62.054	61.541	76.453	89.090
-	-	-	-	-	-
-	-	-	-	-	-
-	-	-	-	-	-
-	-	-	-	-	-
-5.508	-5.508	-5.508	-5.507	-5.505	-5.504

Tabla 5.4: Valor de la media de los resultados de la propuesta MBFOA-RF usando distinto tamaño de paso dinámico.

MBFOA-PA					
$(\frac{G}{GMAX})^4$	$(\frac{G}{GMAX})^2$	$(\frac{G}{GMAX}) - \frac{sin(2\pi(G-GMAX))}{6,3}$	$(\frac{G}{GMAX})$	$(\frac{G}{GMAX})^{1/2}$	$(\frac{G}{GMAX})^{1/4}$
-2.639	-2.541	**-2.707**	-2.258	-2.433	-2.496
-0.144	**-0.149**	-0.146	-0.143	-0.148	-0.141
-0.021	-0.035	-0.020	-0.012	**-0.039**	-0.020
-28646.608	**-28732.964**	-28692.547	-28637.279	-28568.916	-28394.707
-	-	-	-	-	-
-3364.208	-3439.757	-3096.290	**-3518.736**	-3472.409	-3322.408
506.606	1352.590	1526.115	1080.712	1063.056	787.509
0.007	0.004	-0.001	-0.001	**-0.008**	0.001
703319.498	**561992.652**	628098.676	667522.085	689732.766	1129879.810
20894.045	20730.167	20379.848	20969.282	**19598.848**	21480.474
0.874	0.838	0.889	0.880	**0.831**	0.882
-0.802	**-0.806**	-0.765	-0.781	-0.768	-0.786
-	-	-	-	-	-
-	-	-	-	-	-
985.097	987.925	987.133	987.955	**983.661**	984.237
-1.179	**-1.249**	-1.181	-1.164	-1.248	-1.211
-	-	-	-	-	-
-0.221	-0.189	-0.172	-0.192	-0.157	**-0.383**
4662.725	5554.596	5091.255	5796.228	**3862.063**	4894.935
-	-	-	-	-	-
-	-	-	-	-	-
-	-	-	-	-	-
-	-	-	-	-	-
-3.419	**-3.499**	-3.490	-3.227	-3.331	-3.277

Tabla 5.5: Valor de la media de los resultados de la propuesta MBFOA-PFA usando distinto tamaño de paso dinámico.

- MDE es el segundo algoritmo que resuelve más funciones de prueba, solo en las funciones g14, g20 y g22 no logra encontrar al óptimo global.

- APF-GA es el tercer algoritmo que resuelve más funciones de prueba, solo en las funciones g02, g03, g14, g17, g19, g20, g21, g22 y g23 no logra encontrar al óptimo global.

- SR+ES encuentra el óptimo global a las funciones de prueba g01, g04, g06, g08, g09, g11 y g12.

- ATMES es el cuarto algoritmo que más funciones de pruebas resuelve las cuales son g01, g04, g06, g08, g09, g11 y g12.

Parámetros	**Valores**
Población	15
Ciclo quimiotáxico	20
Factor de escalamiento	1.765
Reproducción	2
ξ	1.8
dec	1.02

Tabla 5.6: Conjunto de parámetros utilizados por DyMBFOA en el conjunto de funciones de prueba del CEC2006 y CEC2010.

- SMES es el quinto mejor algoritmo en esta categoría al encontrar al óptimo global en 6 de las 24 funciones de pruebas las cuales son g01, g04, g06, g08, g11 y g12.
- DyMBFOA solo resuelve 10 de las 24 funciones de prueba las cuales son g03, g04, g05, g06, g08, g11, g12, g16, g18 y g24.

Calidad resultados (Con base en la media y al mejor valor encontrado):

- Memetic-SAMSDE es el algoritmo con mejor calidad de resultados en la mayoría de las funciones de prueba, excepto en las funciones g19 y g20.
- El segundo algoritmo con mejor calidad de resultados es MDE, seguido de AP-GA.
- El cuarto algoritmo con mejor calidad de resultados es SR+ES, seguido de ATMES y SMES.
- Por último, DyMBFOA es el algoritmo con menos calidad de resultados.

Funciones de prueba donde se encuentran soluciones factibles:

- Memetic-SAMSDE es el algoritmo que encuentra soluciones factibles a la mayoría de las funciones de prueba, excepto g20.
- MDE y APF-GA no encuentran soluciones factibles a las funciones de prueba g20 y g22.
- DyMBFOA no encuentra soluciones factibles a las funciones de prueba g20, g21 y g22.
- ATMES, SMES y SR+ES no presentan resultados para a las funciones de prueba g14-g24.
- **Resultados de la prueba de signo de Wilcoxon:**
- Existe una diferencia significativa a favor de Memetic-SAMSDE, MDE Y APF-GA al compararlos individualmente con DyMBFOA.
- No existe una diferencia significativa entre los resultados de ATMES, SMES y SR+ES al ser comparados individualmente con DyMBFOA.

En la Tabla 5.9 se presentan los resultados obtenidos por DyMBFOA en términos de tasa de factibilidad y rendimiento exitoso. La propuesta es comparada con otros algoritmos encontrados en el estado del arte.

De acuerdo a los resultados de la Tabla 5.9 se observa que:

Prob.	Criterios	DyMBFOA	Memetic-SAMSDE	APF-GA	MDE	ATMES	SMES	SR+ES
	Evaluaciones	240,000	240,000	500,000	240,000	240,000	240,000	350,000
	Ejecuciones	30	30	25	25	30	30	30
	Wilcoxon		+	+	+	=	=	=
g01	Mejor	-14.0468311297	**-15**	**-15**	**-15**	**-15**	**-15**	**-15**
	Media	-10.11	**-15**	**-15**	**-15**	**-15**	**-15**	**-15**
	Desv.Est.	1.705655	**0**	**0**	**0**	1.6E-014	**0**	**0**
g02	Mejor	-0.738523	**-0.803619**	-0.803601	**-0.8036191**	-0.803339	-0.803601	-0.803515
	Media	-0.570410	**-0.8036191**	-0.803518	-0.78616	-0.790148	-0.785238	-0.781975
	Desv.Est.	0.105	**0**	0.0001	0.012	0.013	0.0167	0.02
g03	Mejor	**-1.0004**	**-1.005**	-1.001	**-1.005**	-1	-1	-1
	Media	-1.0004	**-1.005**	-1.001	**-1.005**	-1	-1	-1
	Desv.Est.	1.598E-005	0	0	**0**	0.000059	0.0000209	0.0000209
g04	Mejor	**-30665.538**	**-30665.539**	**-30665.539**	**-30665.5386**	**-30665.539**	**-30665.539**	**-30665.539**
	Media	-30665.538	**-30665.539**	**-30665.539**	**-30665.5386**	**-30665.539**	**-30665.539**	**-30665.539**
	Desv.Est.	4.666E-005	**0**	0.0001	0.0001	7.4E-012	**0**	0.00002
g05	Mejor	**5126.497**	**5126.497**	**5126.497**	**5126.497**	5126.498	5126.599	**5126.497**
	Media	5194.046	**5126.497**	5127.5423	**5126.497**	5127.648	5174.492	5128.881
	Desv.Est.	110.209543	**0**	1.4324	**0**	1.8	50.06	3.5
g06	Mejor	**-6961.81387**	**-6961.813875**	**-6961.814**	**-6961.814**	**-6961.814**	**-6961.814**	**-6961.814**
	Media	-6961.813833	**-6961.813875**	**-6961.814**	**-6961.814**	**-6961.814**	-6961.284	-6875.94
	Desv.Est.	3.354E-005	**0**	**0**	**0**	4.6E-012	1.85	160
g07	Mejor	24.3425	**24.3062**	**24.3062**	**24.3062**	**24.306**	24.327	24.307
	Media	24.5677	**24.3062**	**24.3062**	**24.3062**	24.316	24.475	24.374
	Desv.Est.	0.130660	**0**	**0**	**0**	0.011	0.132	0.066
g08	Mejor	**-0.095825**	**-0.095825**	**-0.095825**	**-0.095825**	**-0.095825**	**-0.095825**	**-0.095825**
	Media	**-0.095825**	**-0.095825**	**-0.095825**	**-0.095825**	**-0.095825**	**-0.095825**	**-0.095825**
	Desv.Est.	5.762E-018	**0**	**0**	**0**	2.8E-016	**0**	**0**
g09	Mejor	680.63	**680.63**	**680.63**	**680.63**	**680.63**	680.632	**680.63**
	Media	680.66	**680.63**	**680.63**	**680.63**	680.639	680.643	680.656
	Desv.Est.	0.019878	**0**	**0**	**0**	0.01	0.0155	0.034
g10	Mejor	7056.78128	**7049.24802**	**7049.24802**	**7049.24802**	7052.253	7051.903	7051.903
	Media	7143.28334	**7049.24802**	7077.6821	**7049.24802**	7250.437	7253.047	7253.047
	Desv.Est.	69.689763	**0**	51.24	**0**	120	136	136
g11	Mejor	**0.7499**	**0.7499**	**0.7499**	**0.7499**	**0.75**	**0.75**	**0.75**
	Media	0.7499	**0.7499**	**0.7499**	0.7499	**0.75**	**0.75**	**0.75**
	Desv.Est.	5.690E-009	**0**	**0**	**0**	0.00034	0.000152	0.00008
g12	Mejor	**-1**	**-1**	**-1**	**-1**	**-1**	**-1**	**-1**
	Media	**-1**	**-1**	**-1**	**-1**	**-1**	**-1**	**-1**
	Desv.Est.	**0**	**0**	**0**	**0**	0.001	**0**	**0**

Tabla 5.7: 1ra. Parte de las estadísticas básicas y prueba Wilcoxon de los resultados obtenidos por DyMBFOA y otros algoritmos en las funciones de prueba del CEC2006.

- DyMBFOA tiene una tasa de factibilidad del 100 % en la mayoría de los casos de prueba excepto en las funciones g20, g21 y g22 que no obtiene factibilidad en ninguna de sus ejecuciones y en el g23 su tasa es muy pobre.
- DyMBFOA presenta una tasa de éxito del 100 % sólo en las funciones g03, g08, g11, g12 y g24. En las funciones g04, g06, g16 y g18 obtiene una tasa de éxito menor al 100 %. En 15 de las 24 funciones de prueba no tiene éxito, es decir, tiene un 0 % de éxito.
- DyMBFOA tiene en promedio el 83.61 % de tasa de factibilidad la cual es inferior a la alcanzada por los demás algoritmos con los que fue comparado.
- DyMBFOA presenta en promedio una tasa de éxito del 33.33 % muy por debajo de la tasa alcanzada por los demás algoritmos con los que se compara.

El tamaño de paso dinámico en DyMBFOA probado en CEC2010

Los resultados de DyMBFOA probado en las funciones del prueba del CEC2010 con 10 dimensiones son presentados en la Tabla 5.10, Con base en esto se presenta el siguiente análisis:

Funciones de prueba que resuelve al satisfacer la condición $f(\overrightarrow{x}) - f(\overrightarrow{x^*}) \leq 0{,}0001$:

- Memetic-SAMSDE es el algoritmo que más funciones resuelve al encontrar el óptimo global a la mayoría de la función excepto c02.
- ϵDEag es el segundo algoritmo que más funciones resuelve sin embargo no encuentra el óptimo global de las funciones c02, c03, c04, c06, c08 y c17.

Prob.	Criterios	DyMBFOA	Memetic-SAMSDE	APF-GA	MDE	ATMES	SMES	SR+ES
	FES	240,000	240,000	500,000	240,000	240,000	350,000	
	Runs	30	30	25	25	30	30	30
	Wilcoxon		+	+	+	=	=	=
g13	Mejor	0.055473	**0.053942**	**0.053942**	**0.053942**	0.05395	0.053986	0.067543
	Media	0.287508	**0.053942**	**0.053942**	**0.053942**	0.053959	0.166385	0.067543
	Desv.Est.	0.179109	**0**	**0**	**0**	0.000013	0.177	0.031
g14	Mejor	-46.966511	**-47.764888**	-47.76479	-47.764887	-	-	-
	Media	-45.014293	**-47.764888**	-47.76479	-47.764874	-	-	-
	Desv.Est.	0.935908	**0**	0.0001	0.000014	-	-	-
g15	Mejor	961.71517	**961.71502**	**961.71502**	**961.71502**	-	-	-
	Media	961.71527	**961.71502**	**961.71502**	**961.71502**	-	-	-
	Desv.Est.	0.00025	**0**	**0**	**0**	-	-	-
g16	Mejor	**-1.905151**	**-1.905155**	**-1.905155**	**-1.905155**	-	-	-
	Media	-1.904717	**-1.905155**	**-1.905155**	**-1.905155**	-	-	-
	Desv.Est.	0.0005	**0**	**0**	**0**	-	-	-
g17	Mejor	8860.7851	**8853.5397**	8853.5398	**8853.5397**	-	-	-
	Media	8913.8180	**8823.5397**	8888.4876	**8853.5397**	-	-	-
	Desv.Est.	32.6941	**0**	29.0347	**0**	-	-	-
g18	Mejor	**-0.866025**	**-0.866025**	**-0.866025**	**-0.866025**	-	-	-
	Media	-0.865981	**-0.866025**	**-0.865925**	**-0.866025**	-	-	-
	Desv.Est.	6.479E-005	**0**	0.0001	**0**	-	-	-
g19	Mejor	43.346514	32.655593	32.655593	**32.64827**	-	-	-
	Media	55.541448	32.655593	32.655593	33.34125	-	-	-
	Desv.Est.	8.993	**0**	**0**	0.847	-	-	-
g20	Mejor	-	-	-	-	-	-	-
	Media	-	-	-	-	-	-	-
	Desv.Est.	-	-	-	-	-	-	-
g21	Mejor	-	**193.72451**	196.63301	**193.72451**	-	-	-
	Media	-	**193.72451**	199.51581	**193.72451**	-	-	-
	Desv.Est.	-	**0**	2.3565	**0**	-	-	-
g22	Mejor	-	**236.370313**	-	-	-	-	-
	Media	-	**245.738829**	-	-	-	-	-
	Desv.Est.	-	**9.05939**	-	-	-	-	-
g23	Mejor	-108.564914	**-400.0551**	-399.7624	**-400.0551**	-	-	-
	Media	-6.2848	**-400.0551**	-394.7627	**-400.0551**	-	-	-
	Desv.Est.	144.645802	**0**	3.8656	**0**	-	-	-
g24	Mejor	**-5.508013**	**-5.508013**	**-5.508013**	**-5.508013**	-	-	-
	Media	**-5.508013**	**-5.508013**	**-5.508013**	**-5.508013**	-	-	-
	Desv.Est.	8.511E-011	**0**	**0**	**0**	-	-	-

Tabla 5.8: 2da. Parte de las estadísticas básicas y prueba Wilcoxon de los resultados obtenidos por DyMBFOA y otros algoritmos en las funciones de prueba del CEC2006.

Prob.	Tasa de factibilidad								Tasa de éxito							
	EDE	SaDE	MPDE	GDE	JDE-2	MDE	CMODE	DyMBFOA	EDE	SaDE	MPDE	GDE	JDE-2	MDE	CMODE	DyMBFOA
g02	100 %	100 %	100 %	100 %	100 %	100 %	100 %	100 %	100 %	84 %	92 %	72 %	92 %	16 %	100 %	0 %
g03	100 %	100 %	100 %	96 %	100 %	100 %	100 %	100 %	100 %	96 %	84 %	4 %	0 %	100 %	100 %	100 %
g05	100 %	100 %	100 %	96 %	100 %	100 %	100 %	100 %	100 %	100 %	100 %	92 %	68 %	100 %	100 %	0 %
g11	100 %	100 %	100 %	100 %	100 %	100 %	100 %	100 %	100 %	100 %	96 %	100 %	96 %	100 %	100 %	100 %
g13	100 %	100 %	88 %	88 %	100 %	100 %	100 %	100 %	100 %	100 %	48 %	40 %	0 %	100 %	100 %	0 %
g14	100 %	100 %	100 %	100 %	100 %	100 %	100 %	100 %	100 %	80 %	100 %	96 %	100 %	100 %	100 %	0 %
g15	100 %	100 %	100 %	100 %	100 %	100 %	100 %	100 %	100 %	100 %	100 %	96 %	96 %	100 %	100 %	0 %
g17	100 %	100 %	96 %	76 %	100 %	100 %	100 %	100 %	100 %	4 %	28 %	16 %	4 %	100 %	100 %	0 %
g18	100 %	100 %	100 %	84 %	100 %	100 %	100 %	100 %	100 %	92 %	100 %	76 %	100 %	100 %	100 %	87 %
g19	100 %	100 %	100 %	100 %	100 %	100 %	100 %	100 %	100 %	100 %	100 %	88 %	100 %	0 %	100 %	0 %
g20	0 %	0 %	0 %	0 %	4 %	0 %	0 %	0 %	0 %	0 %	0 %	0 %	0 %	0 %	100 %	0 %
g21	100 %	100 %	100 %	88 %	100 %	100 %	100 %	0 %	100 %	60 %	68 %	60 %	92 %	100 %	80 %	0 %
g22	100 %	100 %	0 %	0 %	0 %	0 %	0 %	0 %	0 %	0 %	0 %	0 %	0 %	0 %	0 %	0 %
g23	100 %	100 %	100 %	88 %	100 %	100 %	100 %	7 %	100 %	88 %	100 %	40 %	92 %	100 %	100 %	0 %
Media	**95.83** %	**95.83** %	91 %	88.17 %	91.83 %	91.67 %	91.67 %	83.61 %	91.67 %	83.5 %	84 %	74.17 %	76.67 %	84 %	**95** %	33.33

Tabla 5.9: Comparación de DyMBFOA con otros algoritmos en términos de tasa de factibilidad y tasa de éxito.

- IEMA es el tercer algoritmo que más funciones resuelve aunque sólo encuentra el óptimo global a las funciones c01, c02, c05, c06, c11 y c13.
- DyMBFOA no encuentra el óptimo global a ninguna de las funciones de prueba.

Calidad de los resultados (Con base en la media y al mejor valor encontrado):

- Mememtic-SAMSDE es el algoritmo con mejor calidad de resultados.
- ϵDEag es el segundo algoritmo con mejor calidad al igual que IEMA.
- DyMBFOA es el algoritmo con menos calidad de resultados.

Funciones de prueba donde se encuentran soluciones factibles:

- Todos los algoritmos comparados encuentran soluciones factibles a cada uno de las funciones de prueba.

Resultados de la prueba de signo de Wilcoxon:

- Existe una diferencia significativa a favor de Memetic-SAMSDE, ϵDEag y IEMA al compararlos individualmente con DyMBFOA.

Prob.	Criterios	DyMBFOA	Memetic-SAMSDE	ϵDEag	IEMA
	Evaluaciones	200,000	200,000	200,000	200,000
	Ejecuciones	25	25	25	25
	Wilcoxon		+	+	+
C01	Mejor	-0.747308	**-0.7473104**	**-0.7473104**	**-74731**
	Media	-0.736838	**-0.7473104**	**-0.7470402**	-0.743189
	Desv.Est.	0.01272331	**0**	**0.001323339**	0.00433099
C02	Mejor	-2.2148917	-2.2777099	-2.277702	**-2.27771**
	Media	-2.15302095	-2.2776477	-2.25887	**-2.27771**
	Desv.Est.	0.04522673	0.000059704	2.39E-002	**1.82278E-7**
C03	Mejor	7287.2133447	**0**	**0**	1.47E-016
	Media	36529538066438.2	4.815855E-021	**0**	6.23E-007
	Desv.Est.	100069539918729	6.862013E-021	**0**	1.40E-006
C04	Mejor	0.00195029	**-0.00001**	-9.99E-006	-9.99E-006
	Media	0.19905347	**-0.00001**	-9.92E-006	-9.91E-006
	Desv.Est.	0.32829388	**2.2847E-10**	1.55E-007	8.99E-008
C05	Mejor	46.84194949	**-483.610625**	**-483.6106**	**-483.611**
	Media	258.7935276	**-483.610625**	**-483.6106**	-379.156
	Desv.Est.	100.927966	1.4883E-007	**3.89035E-13**	179.424
C06	Mejor	79.0198985	**-578.66236**	-578.6581	**-578.662**
	Media	274.1803777	**-578.6622**	-578.6528	-551.47
	Desv.Est.	71.182300	**0.0001182591**	3.63E-003	73.5817
C07	Mejor	0.079615	**0**	**0**	1.75E-008
	Media	3.257721	9.30E-024	**0**	3.26E-009
	Desv.Est.	1.735795	1.71E-023	**0**	3.39E-009
C08	Mejor	0.02756	**0**	**0**	1.01E-010
	Media	15.890536	**9.514571E-020**	6.727528	4.0702
	Desv.Est.	36.562941	**3.494602E-019**	5.560648	6.38287
C09	Mejor	5511948544 7.6047	**0**	**0**	1.20E-009
	Media	1487253341388.51	1.29E-021	**0**	1.95E+012
	Desv.Est.	1567458128414.39	2.87E-021	**0**	5.40E+012
C10	Mejor	6865631901 7.7502	**0**	**0**	5.40E-009
	Media	1036502771327.2	7.46E-023	**0**	2.56E+012
	Desv.Est.	821175635669.711	1.56E-022	**0**	3.97E+012
C11	Mejor	-0.001271	**-0.00152271**	**0.001522713**	**-0.00152271**
	Media	-0.000475	**-0.00152271**	**0.001522713**	**-0.001152271**
	Desv.Est.	0.000735	1.36E-008	**6.34103E-11**	2.73E-008
C12	Mejor	-3.917201	**-570.0899**	**-570.0899**	-10.9735
	Media	-0.165891	-33.55340799	**-336.7349**	-0.648172
	Desv.Est.	0.845952	1.16E+002	1.78E+002	**2.19928**
C13	Mejor	-68.429256	**-68.42937**	**-68.42937**	**-68.4294**
	Media	-64.064179	**-67.42937**	-68.42936	-68.0182
	Desv.Est.	2.148450	**0**	1.03E-006	1.40069
C14	Mejor	3.419237	**0**	**0**	8.04E-010
	Media	176.895936	9.78E-021	**0**	56.3081
	Desv.Est.	432.863759	2.10E-020	**0**	182.866
C15	Mejor	556643027001.121	**0**	**0**	9.35E-010
	Media	421335287321 91.8	**2.4762E-020**	1.80E-001	1.58E+008
	Desv.Est.	2145726607 6646.2	**1.0131E-019**	8.81E-001	6.04E+008
C16	Mejor	0.024777	**0**	**0**	4.44E-016
	Media	0.289085	**0**	3.70E-001	0.0330299
	Desv.Est.	0.210234	**0**	3.71E-001	0.0226013
C17	Mejor	0.169658	**0**	1.46E-017	9.48E-015
	Media	3.791325	**8.174E-016**	1.25E-001	0.00315093
	Desv.Est.	2.716678	**1.4492E-015**	1.94E-001	0.0157547
C18	Mejor	0.011253	**0**	3.73E-020	2.24E-015
	Media	15.323537	**3.0395E-025**	9.68E-019	1.62E-014
	Desv.Est.	10.234668	**1.4976E-024**	1.81E-018	3.82E-014

Tabla 5.10: Estadísticas básicas y prueba Wilcoxon de los resultados obtenidos por DyMBFOA y otros algoritmos en el conjunto de funciones de prueba del CEC2010 con 10 dimensiones.

Los resultados obtenido de DyMBFOA en el conjunto de funciones de prueba del CEC2010 con 30 dimensiones son presentados en la Tabla 5.11, Con base en esto se presenta el siguiente análisis:

Funciones de prueba que resuelve al satisfacer la condición $f(\overrightarrow{x}) - f(\overrightarrow{x^*}) \leq 0{,}0001$:

- Memetic-SAMSDE es el algoritmo que más funciones resuelve al encontrar el óptimo global en todas las funciones de prueba.
- ϵDEag sólo resuelve la función c16.

- IEMA y DyMBFOA no encuentra el óptimo global a ninguna de las funciones de prueba.

Calidad de los resultados (Con base en la media y al mejor valor encontrado):

- Mememtic-SAMSDE es el algoritmo con mejor calidad de resultados.
- ϵDEag es el segundo algoritmo con mejor calidad.
- IEMA ocupa el tercer lugar en ésta categoría.
- DyMBFOA es el algoritmo con menos calidad de resultados.

Funciones de prueba donde se encuentran soluciones factibles:

- Mememtic-SAMSDE y ϵDEag encuentran soluciones factibles a todas las funciones de prueba.
- DyMBFOA encuentra soluciones factibles a la mayoría de las funciones, excepto c04 y c11.
- IEMA es el algoritmo que deja sin soluciones factibles a varias funciones las cuales son c03, c04, c11 y c12.

Resultados de la prueba de signo de Wilcoxon:

- Existe una diferencia significativa a favor de Memetic-SAMSDE, ϵDEag y IEMA al compararlos individualmente con DyMBFOA a pesar que éste último (IEMA) no encuentra soluciones factibles a 4 de las 18 funciones de prueba.

En general, DyMBFOA:

- Carece de calidad en los resultados.
- La propuesta logra encontrar en la mayoría de las funciones de pruebas soluciones factibles.
- Las soluciones factible no son soluciones factibles exitosas (superan la condición de éxito de 0.0001).
- La propuesta no encuentra soluciones factibles cuando la función es altamente restringida con condiciones de igualdad.

5.2. MBFOA usando un tamaño de paso adaptativo

Una versión adaptativa para el tamaño de paso también fue probada para conocer el rendimiento de MBFOA-RF con esta tendencia, el nuevo tamaño de paso es basado en la regla de $\frac{1}{5}$ [34] usando la ecuación 5.7:

$$C(i,G+1) = \begin{cases} C(i,G-1)_i SSA & si \quad SR < 0{,}2 \\ C(i,G-1)_i \frac{1}{SSA} & sino \end{cases} \tag{5.7}$$

donde SSA es un parámetro definido por el usuario para controlar el nivel del cambio del valor del tamaño de paso y SR es el número promedio de movimientos exitosos sobre todos los movimientos realizados por todas las bacterias en una G. Un movimiento exitoso es contado cuando una bacteria mejora su solución por la aplicación del operador de nado-giro o agrupamiento [34]. El tamaño de paso inicial es calculado usando la ecuación 5.8:

$$C(i,0) = R(U_k - L_k) \tag{5.8}$$

Prob.	Criterios	DyMBFOA	Memetic-SAMSDE	ϵDEag	IEMA
	Evaluaciones	600,000	60,000	600,000	600,000
	Ejecuciones	25	25	25	25
	Wilcoxon		+	+	+
C01	Mejor	-0.637231	**-0.821884397**	-0.8218255	-0.821883
	Media	-0.439897	-0.815632419	**-0.8208687**	-0.817769
	Desv.Est.	-0.311272	4.41E-003	**0.0007103893**	0.00478853
C02	Mejor	-1.300707	**-2.28809621**	-2.169248	-2.28091
	Media	0.513763	**-2.2777017**	-2.151424	-1.50449
	Desv.Est.	2.541334	**0.0009847005**	1.20E-002	2.14E+000
C03	Mejor	258179733258.962	**1.149732E-020**	2.87E+001	-
	Media	11591256463384.7	**9.85425E-018**	2.88E+001	-
	Desv.Est.	23661845190990.1	**3.47535E-017**	8.05E-001	-
C04	Mejor	-	**-0.000003**	4.70E-003	-
	Media	-	**0.000015**	8.16E-003	-
	Desv.Est.	-	**0.000091**	3.07E-003	-
C05	Mejor	267.653362	**-483.610624**	-453.1307	-286.678
	Media	420.311620	**-483.61058**	-449.546	-270.93
	Desv.Est.	501.748252	**0.000096009**	2.90E+000	14.1169
C06	Mejor	375.928133	**-530.636798**	-528.575	-529.593
	Media	488.178155	**-530.0979007**	-527.9068	-132.876
	Desv.Est.	553.014368	**0.3077**	4.75E-001	561.042
C07	Mejor	19.755250	**5.01729E-025**	1.15E-015	4.82E-010
	Media	43.221967	**9.3441E-020**	2.60E-015	8.49E-010
	Desv.Est.	251.660426	**1.66599E-019**	1.23E-015	4.84E-010
C08	Mejor	20.180497	**3.404908E-021**	2.52E-014	1.12E-009
	Media	57.439258	**1.64526E-017**	7.83E-014	17.7033
	Desv.Est.	438.023380	**3.885606E-017**	4.86E-014	40.8025
C09	Mejor	10315768117110.3	**9.823762E-023**	2.77E-016	7.31E+003
	Media	19595364827259.5	**1.377364E-14**	1.07E+001	2.99E+007
	Desv.Est.	25825455296039.9	**4.586357E-14**	2.82E+001	4.50E+007
C10	Mejor	7486670525457.19	**4.884692E-025**	3.25E+001	2.77E+004
	Media	19601029714587.9	**1.622473E-015**	3.33E+001	1.58E+007
	Desv.Est.	34249076224382.1	**3.642993E-015**	4.55E-001	1.68E+007
C11	Mejor	-	**-0.000392**	-3.27E-004	-
	Media	-	**-0.000391**	-2.86E-004	-
	Desv.Est.	-	**4.797E-7**	2.71E-005	-
C12	Mejor	-0.199225	**-0.1992611**	-0.1991453	-
	Media	-0.156918	**-0.1992558**	3.56E+002	-
	Desv.Est.	0.120461	**2.0089E-6**	2.89E+002	-
C13	Mejor	-65.621976	**-68.42936**	-66.42473	-68.4294
	Media	-61.655339	**-68.2398**	-65.3531	-67.4872
	Desv.Est.	-58.367563	**0.34466**	5.73E-001	0.983662
C14	Mejor	22.937114	**1.568781E-019**	5.02E-014	3.29E-009
	Media	85.378902	2.81E-012	**3.089407E-13**	7.38E-009
	Desv.Est.	394.593107	6.25E-012	**5.608409E-13**	0.307356
C15	Mejor	110682399191877	**7.472138E-021**	2.16E+001	3.12E+004
	Media	188771523049927	**3.940227E-015**	2.16E+001	2.29E+008
	Desv.Est.	286330227956245	**1.306798E-14**	1.10E-004	4.64E+008
C16	Mejor	1.080934	**0**	0	6.16E-012
	Media	1.126333	**0**	2.17E-021	0.00163294
	Desv.Est.	1.162102	**0**	1.06E-020	0.0081647
C17	Mejor	18.108739	**6.906494E-11**	2.17E-001	9.28E-010
	Media	73.794743	**0.0000001229944**	6.33E+000	0.0883974
	Desv.Est.	134.598242	**1.477504E-7**	4.99E+000	0.15109
C18	Mejor	75.299795	**3.003416E-020**	1.23E+000	1.38E-014
	Media	216.882258	**8.360598E-015**	8.75E+001	4.74E-014
	Desv.Est.	467.125260	**3.843566E-14**	1.66E+002	6.57E-014

Tabla 5.11: Estadísticas básicas y prueba Wilcoxon de los resultados obtenidos por DyMBFOA y otros algoritmos en el conjunto de funciones de prueba del CEC2010 con 30 dimensiones.

Esta nueva propuesta lleva por nombre AdMBFOA la cual fue probada en el conjunto de funciones de prueba del CEC2010 con 10 y 30 dimensiones y se comparan sus resultados con MBFOA, DyMBFOA y ϵDEag (este último fue el algoritmo ganador de la competencia del CEC2010). Los parámetros utilizados son presentados en la Tabla 5.12. Fueron llevadas a cabo 25 ejecuciones independientes por cada algoritmo con 199,879 evaluaciones para el conjunto de funciones con 10 dimensiones y 599,908 evaluaciones con 30 dimensiones. Los resultados de cada propuesta son presentados en la Tabla 5.13. De acuerdo con estos resultados se tiene el siguiente análisis:

Con respecto a los problemas con 10 dimensiones. **Funciones de prueba que resuelve al satisfacer la condición $f(\overrightarrow{x}) - f(\overrightarrow{x^*}) \leq 0{,}0001$:**

- Ninguna de las propuestas (MBFOA, DyMBFOA y AdMBFOA) resuelve alguna función de prueba

Calidad de los resultados (Con base en la media y al mejor valor encontrado):

Parámetros	**Valores**
Población	15
Ciclo quimiotáxico	20
Factor de escalamiento	1.765
Reproducción	2
ξ	1.8
dec	1.02
R	0.012
SSA	0.7

Tabla 5.12: Conjunto de parámetros para DyMBFOA en CEC2010.

- AdMBFOA es el mejor algoritmo seguido de DyMBFOA y MBFOA

Funciones de prueba donde se encuentran soluciones factibles:

- MBFOA encuentra soluciones factibles a la mayoría de las funciones excepto C03, C04, C05, C06, C11 y C12
- DyMBFOA encuentra soluciones factibles a todas las funciones de prueba
- AdMBFOA encuentra soluciones factibles a la mayoría de las funciones excepto C02, C11 y C12

Resultados de la prueba de signo de Wilcoxon:

- No existe diferencia significativa al comparar DyMBFOA con MBFOA y DyMBFOA con AdMBFOA
- Existe una diferencia significativa a favor de AdMBFOA al compararlo con MBFOA

Con respecto a las funciones de prueba con 30 dimensiones. **Funciones de prueba que resuelve al satisfacer la condición** $f(\overrightarrow{x}) - f(\overrightarrow{x^*}) \leq 0{,}0001$**:**

- Ninguna de las propuestas (MBFOA, DyMBFOA y AdMBFOA) resuelven alguna función al comparar con el mejor resultado obtenido por ϵDEag (ver columna 5 de la tabla 5.16
- **Calidad de los resultados (Con base en la media y al mejor valor encontrado):**
- AdMBFOA es el mejor algoritmo seguido de DyMBFOA y MBFOA

Funciones de prueba donde se encuentran soluciones factibles:

- MBFOA encuentra soluciones factibles a la mayoría de las funciones excepto al C03, C04, C05, C06, C11 y C12
- DyMBFOA encuentra soluciones factibles a todas las funciones excepto al C04 y C011
- AdMBFOA encuentra soluciones factibles a la mayoría de las funciones excepto al C02 y C11

Resultados de la prueba de signo de Wilcoxon:

- Existe diferencia significativa al comparar DyMBFOA con MBFOA y AdMBFOA con MBFOA (ver Tabla 5.14).

		10 dimensiones			30 dimensiones		
Prob.	**Criterios**	**MBFOA**	**DyMBFOA**	**AdMBFOA**	**MBFOA**	**DyMBFOA**	**AdMBFOA**
C01	Mejor	-0.72913	**-0.74731**	-0.68331	-0.43095	**-0.63723**	-0.40920
	Media	-0.43243	**-0.73684**	-0.44869	-0.33111	**-0.43990**	-0.32818
	Desv.Est.	-0.27579	**0.01272**	-0.29521	-0.25537	-0.31127	**-0.25489**
C02	Mejor	**-2.25763**	-2.21489	-	-0.13025	**-1.30071**	-
	Media	-1.86489	**-2.15302**	-	1.54651	**0.51376**	-
	Desv.Est.	-1.32634	**0.04523**	-	2.69076	**2.54133**	-
C03	Mejor	-	7287.2133	**0.00028**	-	**2.58E+011**	3.48E+013
	Media	-	3.65E+013	**6.20E+010**	-	**1.15E+013**	3.48E+013
	Desv.Est.	-	1.00E+014	**9.05E+011**	-	**2.36E+013**	0
C04	Mejor	-	0.00195	**-0.00001**	-	-	**0.00126**
	Media	-	0.19905	**0.00314**	-	-	**0.32640**
	Desv.Est.	-	0.32829	**0.02182**	-	-	**0.65153**
C05	Mejor	-	46.84195	**-115.82249**	-	267.65336	**240.86653**
	Media	-	258.79353	**58.72780**	-	420.31162	**268.57453**
	Desv.Est.	-	**100.92797**	266.19711	-	501.74825	**312.22061**
C06	Mejor	-	79.01990	**-158.44927**	-	375.92813	**239.37941**
	Media	-	274.18038	**103.72195**	-	488.17816	**333.82931**
	Desv.Est.	-	**71.18230**	369.19021	-	553.01437	**420.46259**
C07	Mejor	11.30504	0.07962	**0.00001**	40.54713	19.75525	**15.16612**
	Media	71.19731	**3.25772**	74.13870	356.17453	**43.22197**	196.28284
	Desv.Est.	318.07288	**1.73580**	431.78739	1541.04851	**251.66043**	1024.42921
C08	Mejor	27.43715	0.02756	**0.00079**	42.19095	20.18050	**8.42260**
	Media	153.99715	**15.89054**	125.52474	185.07248	**57.43926**	399.50976
	Desv.Est.	1261.97777	**36.56294**	448.48592	1510.55716	**438.02338**	3988.41648
C09	Mejor	1.72E+011	5.51E+010	**4285.584**	7.58E+012	**1.03E+013**	1.24E+013
	Media	**6.54E+011**	1.48E+012	1.02E+012	2.13E+013	1.95E+013	**1.76E+013**
	Desv.Est.	1.62E+012	**1.56E+012**	2.16E+012	6.02E+013	2.58E+013	**2.39E+013**
C10	Mejor	3.42E+011	6.86E+010	**3.11E+010**	6.96E+012	7.48E+012	**1.02E+013**
	Media	5.51E+012	**1.03E+012**	1.21E+012	2.08E+013	1.96E+013	**1.65E+013**
	Desv.Est.	2.07E+013	**8.21E+011**	3.92E+012	5.30E+013	3.42E+013	**2.61E+013**
C11	Mejor	-	**-0.00127**	-	-	-	-
	Media	-	**-0.00048**	-	-	-	-
	Desv.Est.	-	**0.00074**	-	-	-	-
C12	Mejor	-	**-3.91720**	-	-	**-0.19923**	-0.19864
	Media	-	**-0.16589**	-	-	-0.15692	**-0.19821**
	Desv.Est.	-	**0.84595**	-	-	0.12046	**-0.19777**
C13	Mejor	-63.45385	**-68.42926**	-65.57840	-61.76138	**-65.62198**	-60.38122
	Media	-60.48999	**-64.06418**	-60.41535	-58.02099	**-61.65534**	-57.38033
	Desv.Est.	-55.84645	**2.14845**	-55.67973	-53.87480	**-58.36756**	-53.91292
C14	Mejor	1666.59875	3.41924	**0.87184**	668.16817	22.93711	**19.03404**
	Media	5149.85245	**176.89594**	156047.8303	3169.01647	**85.37890**	308.33081
	Desv.Est.	15937.05384	**432.86376**	3894159.935	7587.06720	**394.59311**	2894.25070
C15	Mejor	2.01E+013	**5.56E+011**	5.63E+012	1.26E+014	**1.10E+014**	1.54E+014
	Media	6.55E+013	**4.21E+013**	7.15E+013	2.54E+014	**1.88E+014**	2.36E+014
	Desv.Est.	1.35E+014	**2.14E+013**	1.18E+014	3.33E+014	**2.86E+014**	4.33E+014
C16	Mejor	**0.00026**	0.02478	0.06942	**0.03630**	1.08093	1.34E+010
	Media	0.07092	0.28909	**0.06942**	0.45781	1.12633	**0.04387**
	Desv.Est.	0.30646	0.21023	**0.00000**	0.89451	1.16210	**0.15325**
C17	Mejor	0.00465	0.16966	**0.00000**	2.21488	18.10874	**0.06645**
	Media	**0.15870**	3.79133	0.25489	21.17258	73.79474	**2.22721**
	Desv.Est.	**1.12630**	2.71668	1.78423	56.43749	134.59824	**12.51063**
C18	Mejor	0.02635	0.01125	**0.00000**	0.42918	75.29980	**0.00879**
	Media	0.15352	**15.32354**	66.60107	53.65787	216.88226	**7.64652**
	Desv.Est.	**0.40718**	10.23467	861.21096	460.20071	467.12526	**46.59032**

Tabla 5.13: Estadísticas básicas de MBFOA, DyMBFOA y AdMBFOA sobre las funciones de prueba del CEC2010.

- No existe una diferencia significativa al comparar AdMBFOA y DyMBFOA.

10 dimensions			30 dimension		
(AdMBFOA-MBFOA)	(DyMBFOA-MBFOA)	(DyMBFOA-AdMBFOA)	(AdMBFOA-MBFOA)	(DyMBFOA-MBFOA)	(DyMBFOA-AdMBFOA)
+	=	=	=	+	=

Tabla 5.14: Prueba Wilcoxon con 95 % de nivel de significancia sobre los resultados de las funciones de prueba del CEC2010.

La tasa de factibilidad de cada una de las propuestas es presentada en la Tabla 5.15 donde cada una de las propuestas obtiene 100 % de soluciones factibles en cada ejecución en la mayoría de las funciones de prueba. En promedio, DyMBFOA es el algoritmo que más soluciones factibles encuentra con un 90.89 % en las funciones con 10 dimensiones y un 84.89 % en las funciones con 30 dimensiones. AdMBFOA es el segundo algoritmo con mejor promedio de soluciones factibles seguido de MBFOA.

	10 dimensiones			30 dimensiones		
Prob.	MBFOA	DyMBFOA	AdMBFOA	MBFOA	DyMBFOA	AdMBFOA
C01	100	100	100	100	100	100
C02	100	100	0	100	100	0
C03	0	100	96	0	28	4
C04	0	28	28	0	0	8
C05	0	100	84	0	100	12
C06	0	100	100	0	100	100
C07	100	100	100	100	100	100
C08	100	100	100	100	100	100
C09	28	100	100	100	100	100
C10	44	100	100	100	100	100
C11	0	12	0	0	0	0
C12	0	96	0	0	100	8
C13	100	100	100	100	100	100
C14	100	100	100	100	100	100
C15	100	100	100	100	100	100
C16	100	100	4	100	100	100
C17	100	100	28	100	100	100
C18	100	100	80	100	100	100
Media	59.56	90.89	67.78	66.67	84.89	68.44

Tabla 5.15: Tasa de factibilidad de MBFOA, DyMBFOA y AdMBFOA sobre los resultados de las funciones de prueba del CEC2010.

Al comparar cada una de las propuestas con el algoritmo ganador del CEC2010 (ver Tabla 5.16) solo AdMBFOA encuentra mejor resultado que ϵDEag en las funciones C17 y C18. DyMBFOA satisface la condición de éxito en la función C01 y AdMBFOA satisface esta condición en las funciones C04, C07, C17 y c18. MBFOA no satisface en ninguna función la condición de éxito.

En general, AdMBFOA:

- Carece de calidad en los resultados.
- Logra encontrar en la mayoría de las funciones de prueba soluciones factibles.
- No son soluciones factibles exitosas (superan la condición de éxito de 0.0001).
- Una de las desventaja con respecto a la propuesta DyMBFOA es el incremento de dos parámetros a definir por el usuario.

5.3. MBFOA con tamaño de paso estático y aleatorio

Un tamaño de paso aleatorio y estático son probados en MBFOA-RF.

	10 dimensiones				30 dimensiones			
Prob.	**MBFOA**	**DyMBFOA**	**AdMBFOA**	**ϵDEag**	**MBFOA**	**DyMBFOA**	**AdMBFOA**	**ϵDEag**
C01	-0.729128	-0.747308	-0.683306	**-0.7473104**	-0.430954	-0.637232	-0.409200	**-0.821825**
C02	-2.257633	-2.214892	-	**-2.277702**	-0.13025	-1.30071	-	**-2.169248**
C03	-	7287.213345	0.000290	**0**	-	2.58E+011	3.48E+013	**28.67347**
C04	-	0.001950	-0.000010	**-9.9923E-6**	-	-	0.00126	**0.0046981**
C05	-	46.841949	-115.822487	**-483.6106**	-	267.65336	240.86653	**-453.1307**
C06	-	79.019899	-158.449274	**-578.6581**	-	375.92813	239.37941	**-528.575**
C07	11.305040	0.079615	0.000006	**0**	40.54713	19.75525	15.16612	**1.1471E-015**
C08	27.437152	0.027562	0.000793	**0**	42.19095	20.18050	8.42260	**2.5186E-14**
C09	1.7202E+11	5.5119E+10	4285.584281	**0**	7.58E+012	1.03E+013	1.24E+013	**2.7706E-016**
C10	3.4218E+11	6.8656E+10	3.1115E+10	**0**	6.96E+012	7.48E+012	1.02E+013	**32.52002**
C11	-	-0.001272	-	**-0.001522713**	-	-	-	**-3.2684E-04**
C12	-	-3.917201	-	**-570.0899**	-	-0.19923	-0.19864	**-0.1991453**
C13	-63.453854	-68.429257	-65.578396	**-68.42937**	-61.76138	-65.62198	-60.38122	**-66.42473**
C14	1666.598753	3.419238	0.871842	**0**	668.16817	22.93711	19.03404	**5.0158E-14**
C15	2.0196E+13	5.5664E+11	5.6328E+12	**0**	1.26E+014	1.10E+014	1.54E+014	**21.60345**
C16	0.000264	0.024777	0.069418	**0**	0.03630	1.08093	1.34E+010	**0**
C17	0.004649	0.169658	7.6672E-15	**1.46318E-017**	2.21488	18.10874	**6.6446E-02**	0.216571
C18	0.026351	0.011253	7.8898E-17	**3.73143E-020**	0.42918	75.29980	**8.7939E-03**	1.226054

Tabla 5.16: MBFOA, DyMBFOA y AdMBFOA comparado ϵDEag algoritmo ganador en la competencia del CEC2010.

El tamaño de paso estático es un valor aleatorio $\in (0, 0.01]$ fijo durante la búsqueda. Por otra parte, el tamaño de paso aleatorio es un valor entre (0,1) generado en cada generación del algoritmo.

Ambas propuestas son comparadas con los resultados de la propuesta adaptativa y dinámica con el objetivo de conocer que tamaño de paso mejora el rendimiento del algoritmo.

Las propuestas de tamaño de paso aleatorio y estático fueron probadas en el conjunto de funciones de prueba del CEC2006 8.1. Los resultados obtenidos por ambas propuestas se comparan contra los resultados obtenidos por las propuestas de tamaño de paso adaptativa y dinámica. El valor de los parámetros utilizados por el algoritmo son presentados en la Tabla 5.17.

parameter	**Valor**
s_b	20
n_c	20
s_r	1
β	1.5
$RepCycle$	100
$gmax$	valor para llegar a 240,000 evaluaciones
ξ	1.2
dec	1.08

Tabla 5.17: Conjunto de parámetros para MBFOA con tamaño de paso aleatorio y estático.

Los resultados obtenidos por cada una de las propuestas son presentados en la Tabla 5.18 y comparados con dos algoritmos del estado del arte, uno variante de ED y otro una mejora de PSO. Con base en los resultados obtenidos por la propuesta se obtiene el siguiente análisis:

Funciones de prueba que resuelve al satisfacer la condición $\overrightarrow{f(x)} - \overrightarrow{f(x^*)} \leq 0{,}0001$:

- Memetic-SAMSDE resuelve casi todas las funciones de prueba excepto g05 y g20.
- IPSO resuelve solo 9 de las 24 funciones de prueba las cuales son g01, g04, g06, g08, g11, g12, g15, g16 y g24.
- DyMBFOA resuelve solo 8 de las 24 funciones de prueba las cuales son g04, g06, g11, g12, g15, g18 y g24.

- rMBFOA resuelve solo 6 de las 24 funciones de prueba las cuales son g03, g08, g11, g12, g15 y g24.
- sMBFOA resuelve solo 5 de las 24 funciones de prueba las cuales son g03, g08, g11, g15 y g24.
- AdMBFOA resuelve solo 4 de las 24 funciones de prueba las cuales son g08, g11, g12 y g24.

Calidad de resultados (Con base en la media y al mejor valor encontrado):

- Memetic-SAMSDE es el algoritmo con mejor calidad de resultados en la mayoría de las funciones de prueba, excepto en las funciones g19 y g20.
- El segundo algoritmo con mejor calidad de resultados es IPSO.
- El tercer algoritmo con mejor calidad de resultados es DyMBFOA seguido de rMBFOA, sMBFOA y AdMBFOA.

Funciones de prueba donde se encuentran soluciones factibles:

- Memetic-SAMSDE es el algoritmo que encuentra soluciones factibles a la mayoría de las funciones de prueba, excepto g20.
- IPSO es el segundo algoritmo que encuentra soluciones factibles a la mayoría de las funciones de prueba, excepto g20 y g22.
- rMBFOA, sMBFOA y DyMBFOA encuentran soluciones factibles a la mayoría de las funciones de prueba, excepto g20, g21 y g22.
- AdMBFOA solo encuentra soluciones factibles a 16 de las 24 funciones de prueba.
- **Resultados de la prueba de signo de Wilcoxon:**
- Existe una diferencia significativa a favor de rMBFOA al ser comparado contra sMBFOA, sin embargo, no existe una diferencia significativa al ser comparado contra DyMBFOA y AdMBFOA.
- No existe una diferencia significativa entre los resultados de DyMBFOA y AdMBFOA, pero ambos algoritmos presentan una diferencia significativa a favor al ser comparados contra sMBFOA.
- Existe una diferencia significativa a favor de Memetic-SAMSDE al compararlo contra DyMBFOA, por lo tanto, también lo es sí se compara contra AdMBFOA, sMBFOA y rMBFOA.
- Existe una diferencia significativa a favor de DyMBFOA al compararlo contra IPSO.

En la tabla 5.18 una comparación de los resultados finales de cada variante de MBFOA y de Memetic-SAMSDE e IPSO son presentados. Memetic-SAMSDE es una propuesta de varias variantes de Evolución diferencial combinadas con un algoritmo de búsqueda local para resolver CNOPs. DyMBFOA fue competitivo en ocho problemas de prueba. Sin embargo, Memetic-SAMSDE tuvo mejores resultados, ya que utiliza SQP, un poderoso método para resolver CNOPs, como buscador local cada 50 generaciones.

En la tabla 5.19 se presentan la tasa de factibilidad obtenida por cada variante de MBFOA. Aparte de problema g23, DyMBFOA obtuvo las mejores tasas de factibilidad entre las cuatro variantes, seguido por rMBFOA, sMBFOA y AdMBFOA como el peor de ellos.

Prob.	Criterios	rMBFOA	sMBFOA	DyMBFOA	AdMBFOA	Memetic-SAMSDE	IPSO
	Ejecuciones	25	25	25	25	25	30
	FEs	239,267	239,200	239,200	239,200	240,000	160,000
g01	Mejor	-12.71	-11.42	-14.99	-14.95	**-15**	**-15**
-15	Media	-9.0557	-7.6267	-14.9133	-14.8271	**-15**	**-15**
	Desv.Est.	2.12E+00	2.20E+00	9.01E-02	8.30E-02	**0**	0.00E+00
g02	Mejor	-0.7052681	-0.3795004	-0.8035107	-0.7346371	**-0.8036191**	-0.802629
-0.8036191	Media	-0.5508	-0.3102	-0.7093	-0.68311	**-0.8036191**	-0.713879
	Desv.Est.	5.87E-02	3.70E-02	5.19E-02	**2.51E-02**	**0**	4.62E-02
g03	Mejor	**-1.00047**	**-1.00048**	-0.99143	-0.9992	**-1.005**	-0.641
-1.0005	Media	-1.0004	-1.0003	-0.82	-0.7927	**-1.005**	-0.154
	Desv.Est.	**0**	2.00E-04	1.28E-01	1.68E-01	**0**	1.70E-01
g04	Mejor	-30662.345	-30593.6367	**-30665.5387**	-30655.99671	**-30665.539**	**-30665.539**
-30665.5387	Media	-30378.8117	-30196.618	-30665.5384	-30628.05037	**-30665.539**	**-30665.539**
	Desv.Est.	2.91E+02	2.53E+02	4.00E-04	1.39E+01	**0**	7.40E-12
g05	Mejor	5137.3697	5128.6938	5134.8775	-	**5126.497**	5126.502
5126.4967	Media	5137.3697	5128.6938	5557.7082	-	**5126.497**	5135.521
	Desv.Est.	**0**	**0**	3.48E+02	-	**0**	1.23E+01
g06	Mejor	-6961.8124	-6961.79248	**-6961.81387**	-6961.03051	**-6961.813875**	**-6961.814**
-6961.81388	Media	-6961.7912	-6961.7142	-6961.8139	-6955.090598	**-6961.813875**	**-6961.814**
	Desv.Est.	1.09E-02	6.84E-02	1.45E-05	3.71E+00	**0**	2.81E-05
g07	Mejor	32.2645	144.2565	24.3960	25.0482	**24.3062**	24.366
24.3062	Media	561.6848	1277.4473	24.7146	42.2303	**24.3062**	24.691
	Desv.Est.	6.75E+02	1.03E+03	2.47E-01	1.28E+01	**0**	2.20E-01
g08	Mejor	**-0.095825**	**-0.095825**	**-0.095825**	**-0.09581**	**-0.095825**	**-0.095825**
-0.095825	Media	-0.0958	-0.0888	-0.0958	-0.0957	**-0.095825**	**-0.095825**
	Desv.Est.	3.22E-10	1.01E-02	9.40E-18	1.00E-04	**0**	4.23E-17
g09	Mejor	682.97	746.56	680.68	680.68	**680.63**	680.638
680.63	Media	1851.3353	6994.1909	680.6821	692.2547	**680.63**	680.674
	Desv.Est.	5.13E+03	2.78E+04	3.39E-02	5.88E+00	**0**	3.00E-02
g10	Mejor	9164.86299	10465.11966	8002.0291	7322.630605	**7049.24802**	7053.963
7049.24802	Media	13304.969	14664.8471	10557.9725	9169.400	**7049.24802**	7306.466
	Desv.Est.	3.21E+03	3.24E+03	1.44E+03	1.13E+03	**0**	2.22E+02
g11	Mejor	**0.7499**	**0.7499**	**0.7499**	**0.7499**	**0.7499**	**0.7499**
0.7499	Media	**0.7499**	**0.7499**	**0.7499**	**0.7499**	**0.7499**	0.753
	Desv.Est.	1.47E-07	9.38E-05	7.65E-07	1.00E-04	**0**	6.53E-03
g12	Mejor	**-1**	-0.9998	**-1**	**-0.9999**	**-1**	**-1**
-1	Media	-0.9966	-0.993	**-1**	-0.9999	**-1**	**-1**
	Desv.Est.	3.10E-03	6.70E-03	**0**	1.48E-06	**0**	0.00E+00
g13	Mejor	0.055784	0.086864	0.081754	-	**0.053942**	0.066845
0.053941	Media	0.4269	1.7568	0.408	-	**0.053942**	0.430408
	Desv.Est.	2.81E-01	2.77E+00	1.61E-01	-	**0**	2.30E+00
g14	Mejor	-47.21327	-47.5851	-46.327887	-	**-47.764888**	-47.449
-47.764888	Media	-45.584	-45.1138	-44.0341	-	**-47.764888**	-44.572
	Desv.Est.	9.57E-01	1.63E+00	1.43E+00	-	**0**	1.58E+00
g15	Mejor	**961.71504**	**961.71509**	**961.71506**	-	**961.71502**	**961.715**
961.71502	Media	**961.9617**	964.0536	961.8524	-	**961.71502**	962.242
	Desv.Est.	6.19E-01	2.68E+00	4.09E-01	-	**0**	6.20E-01
g16	Mejor	-1.901861	-1.882024	-1.904699	-1.895311	**-1.905155**	**-1.905**
-1.905155	Media	-1.7469	-1.6121	-1.8064	-1.8647	**-1.905155**	**-1.905**
	Desv.Est.	1.16E-01	1.82E-01	7.14E-02	1.79E-02	**0**	2.42E-12
g17	Mejor	8871.66724	8971.787	8864.4728	-	**8853.5397**	8863.293
8853.53967	Media	9032.6511	8973.1112	9036.7973	-	**8823.5397**	8911.738
	Desv.Est.	1.41E-02	1.87E+00	1.21E+02	-	**0**	2.73E+01
g18	Mejor	-0.865638	-0.759252	**-0.866025**	-0.856673	**-0.866025**	-0.865994
-0.866025	Media	-0.6921	-0.4403	-0.8656	-0.5239	**-0.866025**	-0.862842
	Desv.Est.	1.58E-01	1.45E-01	7.00E-04	1.88E-01	**0**	4.41E-03
g19	Mejor	50.87558	73.783047	36.585541	44.672686	**32.655593**	33.967
32.655592	Media	79.2473	223.2578	46.631	56.8306	**32.655593**	37.927
	Desv.Est.	2.90E+01	1.07E+02	6.30E+00	9.44E+00	**0**	3.20E+00
g20	Mejor	-	-	-	-	-	-
0.204979	Media	-	-	-	-	-	-
	Desv.Est.	-	-	-	-	-	-
g21	Mejor	-	-	-	-	**193.72451**	193.758
193.72451	Media	-	-	-	-	**193.72451**	217.356
	Desv.Est.	-	-	-	-	**0**	2.65E+01
g22	Mejor	-	-	-	-	**236.370313**	-
236.430975	Media	-	-	-	-	**245.738829**	-
	Desv.Est.	-	-	-	-	**9.05939**	-
g23	Mejor	-214.508	-44.9945	-69.3804	-0.0471	**-400.0551**	-250.707
-400.0551	Media	-9.4519	1.3133	4.6749	-0.0282	**-400.0551**	-99.598
	Desv.Est.	9.80E+01	2.28E+01	6.06E+01	1.18E-02	**0**	1.20E+02
g24	Mejor	**-5.508011**	**-5.507963**	**-5.508013**	**-5.508013**	**-5.508013**	**-5.508**
-5.508013	Media	-5.4609	-4.9452	**-5.508**	**-5.508**	**-5.508013**	**-5.508**
	Desv.Est.	1.60E-01	4.16E-01	3.10E-11	9.06E-16	**0**	9.03E-16

Algoritmos	Criterio	p-value	Decision
rMBFOA-sMBFOA	Mejor valor de aptitud	3.74E-03	+
rMBFOA-DyMBFOA	Mejor valor de aptitud	5.23E-02	=
rMBFOA-AdMBFOA	Mejor valor de aptitud	2.34E-01	=
sMBFOA-DyMBFOA	Mejor valor de aptitud	6.14E-03	/
sMBFOA-AdMBFOA	Mejor valor de aptitud	1.39E-02	/
DyMBFOA-AdMBFOA	Mejor valor de aptitud	7.11E-01	=
DyMBFOA-Memetic-SAMSDE	Mejor valor de aptitud	1.77E-02	/
DyMBFOA-IPSO	Mejor valor de aptitud	2.71E-02	+

Tabla 5.18: Resultados obtenido por las cuatro variantes de MBFOA y otros algoritmos.

La Tabla 5.20 incluye la tasa de éxito obtenido por cada variante de MBFOA, éstas son capaces de alcanzar una buena aproximación de la solución óptima factible (basado en los resultados de la Tabla 5.18), está claro que tales soluciones no satisfacen la tolerancia utilizada en este experimento. La variante más competitiva con respecto a la tasa de éxito fue DyMBFOA.

A fin de analizar el efecto de los cuatro mecanismos de control de tamaño de paso, el número de nados exitosos y la tasa de nados exitosos fueron calculados en dos funciones de prueba representativas $g03$ y $g20$. Ambos funciones tienen una región factible diminuta. Sin embargo, las cuatro variantes obtienen una tasa factible 100 % en $g03$ y 0 % en $g20$.

Prob.	rMBFOA	sMBFOA	DyMBFOA	AdMBFOA
g01	100	100	100	100
g02	100	100	100	100
g03	100	100	100	100
g04	100	100	100	100
g05	4	4	80	-
g06	100	100	100	100
g07	100	100	100	100
g08	100	100	100	100
g09	100	100	100	100
g10	100	96	100	100
g11	100	100	100	100
g12	100	100	100	100
g13	100	76	100	-
g14	100	100	100	-
g15	100	100	100	-
g16	100	100	100	100
g17	20	8	68	-
g18	96	88	100	100
g19	100	100	100	100
g20	-	-	-	-
g21	-	-	-	-
g22	-	-	-	-
g23	80	64	32	36
g24	100	100	100	100

Tabla 5.19: Tasa de factibilidad obtenida por cada variante de MBFOA.

Prob.	rMBFOA	sMBFOA	DyMBFOA	AdMBFOA
g01	-	-	-	-
g02	-	-	-	-
g03	92	40	-	-
g04	-	-	60	-
g05	-	-	-	-
g06	-	-	100	-
g07	-	-	-	-
g08	100	56	100	48
g09	-	-	-	-
g10	-	-	-	-
g11	100	92	100	36
g12	28	-	100	100
g13	-	-	-	-
g14	-	-	-	-
g15	12	4	8	-
g16	-	-	-	-
g17	-	-	-	-
g18	-	-	44	-
g19	-	-	-	-
g20	-	-	-	-
g21	-	-	-	-
g22	-	-	-	-
g23	-	-	-	-
g24	88	8	100	100

Tabla 5.20: Tasa de éxito obtenida por cada variante de MBFOA.

Los nados exitosos de cada generación de la ejecución localizada en la mediana de las 25 ejecuciones independientes reportadas en la Tabla 5.18 son presentados en la Figura 5.1 para la función de prueba $g03$ y en la Figura 5.2 para la función de prueba $g20$. Los resultados del original MBFOA son incluidos como una referencia [78].

La tasa de nados exitosos en la función $g03$ para rMBFOA, sMBFOA, DyMBFOA y AdMBFOA es 11,24 %, 9,68 %, 11,40 % y el 14,59 %, respectivamente. En cuanto a $g20$, los valores fueron 6,91 %, 4,42 %, 21,67 % y el 19,70 % para rMBFOA, sMBFOA, dMBFOA y aMBFOA, respectivamente.

Es importante remarcar, de las figuras 5.1 y 5.2, la capacidad de DyMBFOA para generar nados con éxito al final del proceso de búsqueda en ambas funciones de prueba. Este comportamiento permite a esta variante de MBFOA encontrar mejores soluciones con respecto a las otras variantes (como se informa en la Tabla 5.18), las cuales tuvieron un comportamiento de muchos nados de éxito al principio de la búsqueda y muy pocos al final.

Para analizar el efecto de los mecanismos de control de tamaño del paso para llegar a la región fac-

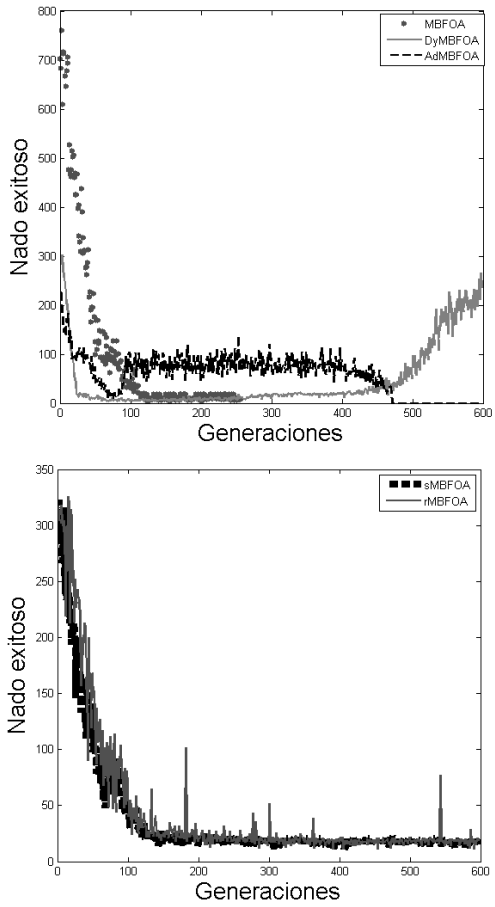

Figura 5.1: Nados exitosos por cada generación de las cuatro variantes de MBFOA en la función $g03$.

tible, el promedio de la suma de violación de restricciones por generación del algoritmo fue medido en la ejecución localizada en el valor de la mediana de las 25 ejecuciones independientes. Las cuatro variantes de MBFOA fueron probadas en la función de prueba $g20$ (donde todas las variantes fueron incapaces de generar soluciones factible). El MBFOA original se incluye como referencia [78]. La Figura 5.3 muestra esta información la cual sugiere que los cuatro mecanismos de control de tamaño de paso son capaces de mejorar, de una manera similar, el acercamiento a la región factible con respecto al original MBFOA. Este comportamiento puede conducir a considerar operadores especiales [108] o búsqueda local [109] para favorecer la generación de soluciones factible.

Para las funciones de prueba $g03$ y $g20$ el comportamiento de AdMBFOA se representa en La Figura 5.4, donde se muestra el valor del tamaño de paso por generación del algoritmo en la ejecución localizada en el valor de la mediana de las 25 ejecuciones independientes realizadas. Para $g03$, donde el comportamiento de AdMBFOA fue bueno, se detecta sólo un aumento de valor de tamaño de paso antes de la generación 100. Después de eso, el tamaño de paso es un valor muy pequeño. Por otro lado, para el problema de prueba $g20$, donde la región factible fue casi alcanzada, los valores de tamaño de paso más grandes se presentan al principio de las generaciones y un pequeño incremento es visto antes de la generación 50 del algoritmo. Después de eso, el valor del tamaño de paso son valores cercanos a cero. El comportamiento antes mencionado, junto con el pobre rendimiento de

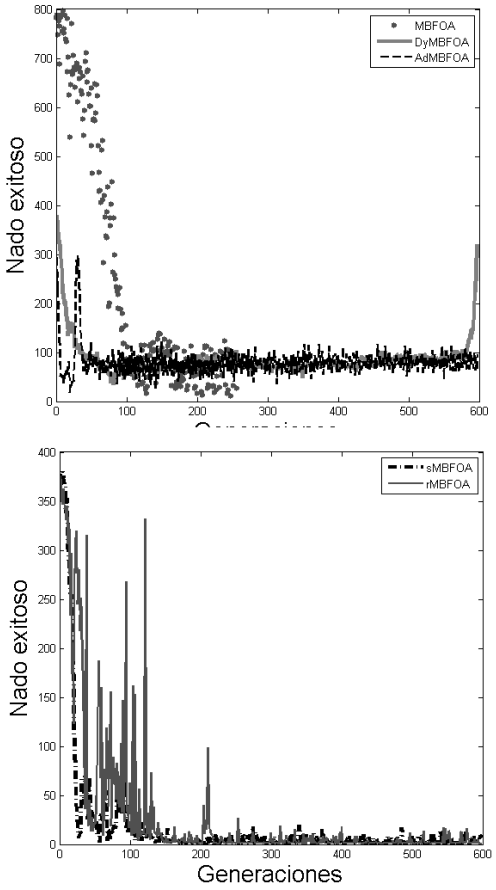

Figura 5.2: Nados exitosos por cada generación de las cuatro variantes de MBFOA en la función $g20$.

AdMBFOA mostrado en los resultados de la Tabla 5.18, sugieren la incapacidad del mecanismo de control de tamaño de paso adaptativo para evitar la convergencia prematura.

En general:

- Los resultados de la prueba de confianza Wilcoxon no encontro una diferencia significativa entre las variantes rMBFOA, DyMBFOA y AdMBFOA.
- Los resultados de DyMBFOA tienen una mejor calidad basados en la media y el mejor valor de aptitud encontrado, además, es la propuesta que más funciones de prueba resuelve.
- Aunque los resultados de DyMBFOA no son altamente competitivos con el algoritmo Memetic-SAMSDE, el comportamiento de esta propuesta puede conducir a considerar operadores especiales o un buscador local para favorecer la generación de soluciones factibles.

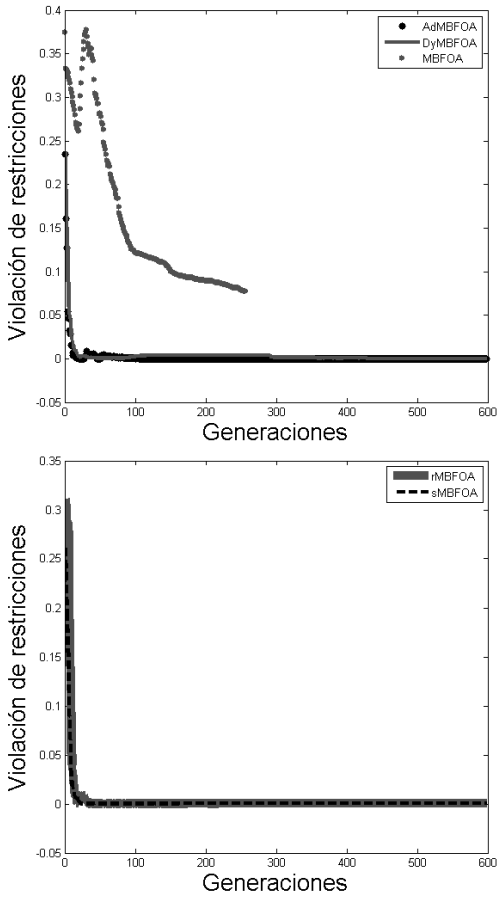

Figura 5.3: Suma de violación de restricciones por generación de cada variante de MBFOA en la función de prueba $g20$.

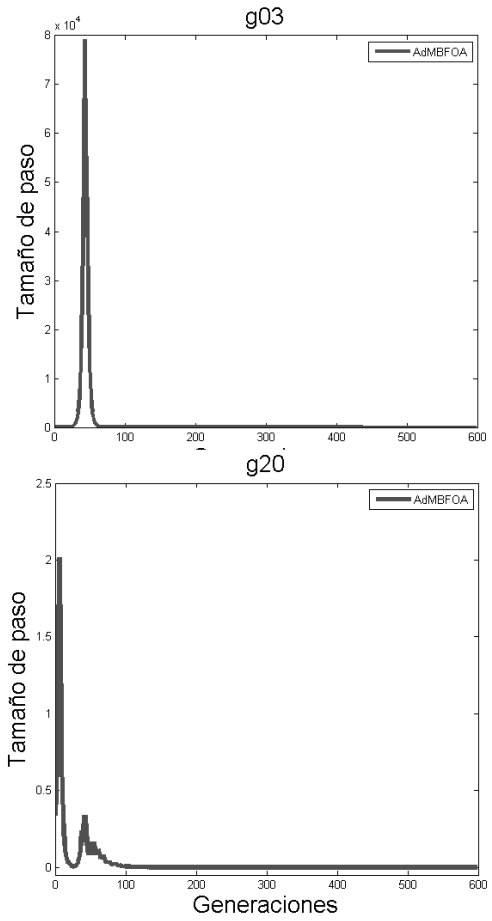

Figura 5.4: Comportamiento de AdMBFOA en la función de prueba $g03$ y $g20$.

Capítulo 6

IMBFOA

En este capítulo se describen las modificaciones realizadas a MBFOA para mejorar su rendimiento en PONR, la propuesta es llamada IMBFOA (por sus siglas en Inglés *Improved Modified Bacterial Foraging Optimization Algorithm*) la cual implementa un mecanismo de sesgo para crear la población inicial, dos operadores de nado, un operador diferente de dirección aleatoria, tamaños de paso dinámico y un buscador local.

6.1. Mecanismos de IMBFOA

En esta sección se describen los mecanismos propuestos para mejorar el rendimiento de MBFOA y el pseudocódigo derivado.

6.1.1. Mecanismo de sesgo

El mecanismo de sesgo para crear la población inicial de bacterias está motivado por el creciente interés en las técnicas de inicialización de algoritmos inspirados en la naturaleza [110]. La población inicial de S_b se integra por tres grupos. El primer grupo está integrado con bacterias aleatorias sesgadas al límite inferior de las variables de decisión. El segundo grupo está integrado con bacterias aleatorias sesgadas al límite superior de las variables de decisión. Por último, un grupo de bacterias situadas aleatoriamente sin sesgo, como en el MBFOA original, es generado. Las fórmulas para fijar los límites para el primer y segundo grupo por variable de decisión se presentan en las Ecuaciones 6.1 y 6.2.

$$[L_i, L_i + ((U_i - Li)/ss)] \tag{6.1}$$

$$[U_i - ((U_i - Li)/ss), U_i] \tag{6.2}$$

donde ss es el tamaño de sesgo ($ss > 1$), los valores grandes aumentan el efecto de inclinación, de lo contrario, disminuye el efecto de sesgo. La figura 6.1 muestra un ejemplo con una población de 30 bacterias, en un espacio de búsqueda bidimensional con $-5 \leq x_i \leq 5$ $(i = 1, 2)$. Considerando $ss = 8$, el primer grupo se genera entre $[-5, -3, 75]$ y el segundo grupo entre [3,75, 5] para ambas variables de decisión, ya que tienen el mismo rango. En la figura RbLx son las bacterias aleatorias sesgadas al límite inferior de las variables de decisión, RbUx son las bacterias aleatorias sesgadas al límite superior de las variables de decisión y RbBx son bacterias aleatorias dentro de los límites de las variables de decisión.

El objetivo de este sesgo en la población inicial, combinada con los dos operadores de nado y el tamaño de paso aleatorio, es evitar que el algoritmo converja prematuramente (comportamiento observado en el MBFOA original, causado por su población, el proceso de reproducción y el tamaño de paso fijo), y mejorar la exploración y explotación del espacio de búsqueda en la fase inicial de la búsqueda.

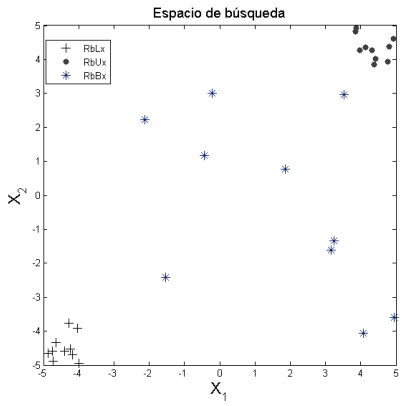

Figura 6.1: Población inicial de bacterias usando el mecanismo de sesgo.

6.1.2. Operador de dirección aleatoria

Otra de las modificaciones realizadas al algoritmo es que el rango de direcciones aleatorias usadas como movimiento de giro de las bacterias generado con la ecuación 3.6 es disminuido. Originalmente el rango es entre [-1,1], sin embargo, se observó en ejecuciones realizadas al algoritmo que un rango menor favorece la búsqueda en problemas altamente restringidos que tienen una región factible muy pequeña. El usuario final debe determinar que rango prefiere de acuerdo a la complejidad del problema a resolver en los parámetros υ y τ donde $-1 \leq \upsilon \leq 0, 0 \leq \tau \leq 1$ y $\upsilon < \tau$.

6.1.3. Operadores de nado en el proceso quimiotáxico

Un nado exploratorio es propuesto el cual será intercalado en el proceso quimiotáxico con el operador de nado original calculado con la ecuación 6.4. En el ciclo medio del proceso qumiotáxico se lleva a cabo el *swarming* con la ecuación 3.4. El nado exploratorio, a diferencia del nado original, permite a las bacterias hacer nados más largos, el cual está definido en la siguiente Ecuación 6.3:

$$\theta^i(j+1, G) = \theta^i(j, G) + \phi(i) \tag{6.3}$$

Si la nueva posición de la bacteria es mejor que la previa, entonces está será la nueva posición de la bacteria y otro nado en la misma dirección será realizado en el siguiente ciclo quimiotáxico. De lo contrario un giro es computado con el otro operador de nado.

6.1.4. Tamaño de paso dinámico

El nado original o de explotación es calculado con la ecuación 6.4

$$\theta^i(j+1, G) = \theta^i(j, G) + C(i, G)\phi(i) \tag{6.4}$$

donde $C(i,G)$ es el vector de tamaño de paso dinámico inspirado en en las fórmulas presentadas anteriormente en el capítulo 5 y computado con la ecuación 6.5. Cada valor K del vector decrece dinamicamente en cada generación del algoritmo.

$$C(i,G+1)_k = C(i,G)_k \frac{G}{GMAX}, k = 1, ..., n \tag{6.5}$$

$C(i,G+1)_k$ es el vector con los nuevos valores de k que corresponden a las variables de diseño del problema y $GMAX$ es el número máximo de generaciones del algoritmo. El valor inicial de $C(i,G)$ es calculado como se indica en la ecuación 3.1 pero el parámetro R ya no es usado.

Es importante mencionar que el nado de exploración permite nados más grandes, caso contrario del nado de explotación. También es importante observar que no siempre se intercalarán los nados de exploración y explotación, puesto que si una bacteria no mejora su posición con el nado (sea de exploración o explotación) un giro será realizado por la bacteria. El ciclo quimiotáxico terminará hasta cumplir con el número máximo de ciclos N_c.

6.1.5. Buscador local

En esta propuesta se integra a MBFOA uno de los métodos de programación matématica más exitosos llamado Sequential Quadratic Programming (SQP) [111] como buscador local con el objetivo de acercar o introducir a las bacterias dentro de la región factible o sacar de óptimos locales a las bacterias y moverlas a otro lugar dentro de la región factible. En este trabajo es utilizado SQP encontrado en la librería de Matlab R2009b [112].

La propuesta tiene una estructura simple (ver figura 7.5), donde SQP es usado solo en la primera y a la mitad de las generaciones del algoritmo después de los procesos: quimiotáxico, agrupamiento, reproducción y eliminación-dispersión con la información de la mejor bacteria de la población de la generación actual, sin embargo, SQP puede ser usado en cualquier generación del algoritmo (definido por el usuario en el parámetro SQP_G). En esta propuesta, estas dos generaciones son seleccionada para usar SQP porque se desea introducir a las bacterias en la región factible en la primera generación y dejar que las bacterias exploren y exploten esta región en la mayoría de las generaciones. A esta propuesta le llamaremos IMBFOA (por sus siglas en Inglés).

Uno de los inconvenientes de usar SQP es que en sus iteraciones ocupa información de segundo orden, por lo que dependiendo del tipo de función objetivo y/o restricciones algunos valores se pueden indefinir [113]. Para las funciones que presentan logaritmos MBFOA no hace uso del SQP.

El pseudocódigo de IMBFOA es presentado en el algoritmo 2. En **Negritas** son remarcados los cambios con respecto a MBFOA. Los parámetros de entrada son el número de bacterias S_b, límite del ciclo quimiotáxico N_c, número de bacterias a reproducir S_r, $RepCycle$ frecuencia de reproducción, factor de escalamiento β, porcentaje de tamaño de paso inicial R, SQP_G frecuencia del buscador local, υ rango de dirección menor, τ rango de dirección mayor y el número de generaciones $GMAX$.

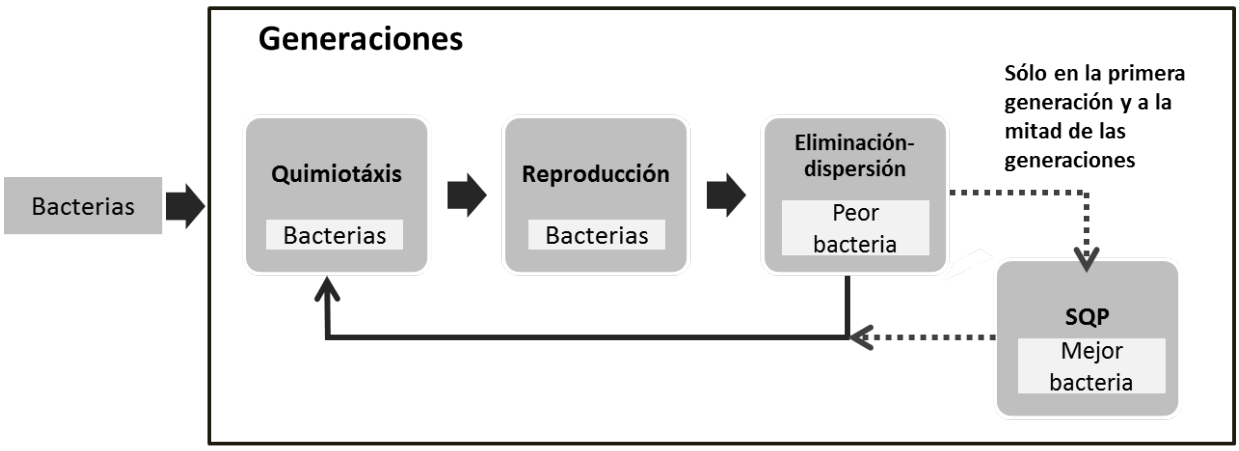

Figura 6.2: Procesos de IMBFOA.

Crear una población inicial de bacterias usando el mecanismo de sesgo $\theta^i(j,0)\ \forall i,\ i = 1,\ldots,S_b$
Evaluar $f(\theta^i(j,0))\ \forall i,\ i = 1,\ldots,S_b$
for *G=1 to GMAX* **do**
  for *i=1 to* S_b **do**
    for *j=1 to* N_c **do**
      Realizar el proceso quimiotáxico con la Ec. 3.6 entre [υ,τ] y la Ec. 6.4 y el operador atrayente 3.4 para la bacteria $\theta^i(j,G)$ considerando la técnica de manejo de restricciones
    end
  end
  if *G mod RepCycle == 0* **then**
    Realizar el proceso de reproducción ordenando las bacterias Con base en la técnica de manejo de restricciones, eliminar a las peores bacterias $S_b - S_r$ y duplicar a las mejores.
  end
  Realizar el proceso de eliminación-dispersión eliminando a la peor bacteria $\theta^w(j,G)$ de la población actual considerando la técnica de manejo de restricciones.
  Actualizar el tamaño de paso dinámicamente usando $(\frac{G}{GMAX})$
  if *G mod* LS_G *==0* **then**
    Aplicar SQP a la mejor bacteria de la población. Si la bacteria obtenida es mejor, ésta reemplaza a la mejor bacteria.
  end
end

Algoritmo 2: Pseudocódigo de IMBFOA.

6.2. IMBFOA en problemas de prueba

IMBFOA fue probado en las funciones de prueba de los benchmark CEC2006 y CEC2010 resumidad en el Anexo 8. SQP fue configurado en Matlab 2009b para realizar máximo 5000 evaluaciones en su intervención dando prioridad al algoritmo de bacterias de explorar el espacio de búsqueda, también la tolerancia de violación de restricciones de igualdad fue ajustada a 1E-04. Los resultados obtenidos son comparados con algunos algoritmos del estado del arte donde los parámetros utilizados por IMBFOA son presentados en la Tabla 6.1.

Los resultados obtenidos por IMBFOA en los problemas de prueba del CEC2006 son presentados en

Parámetro	Valor
S_b	20
N_c	24
S_r	1
β	1.5
$RepCycle$	100
$GMAX$	Valor para alcanzar el máximo número de evaluaciones
SQP_G	1 y $(GMAX/2)$ generación
τ	-0.25
υ	0.15

Tabla 6.1: Valor de los parámetros a usar por MBFOA con SQP.

la Tabla 6.2.

Funciones de prueba que resuelve al satisfacer la condición $f(\overrightarrow{x}) - f(\overrightarrow{x^*}) \leq 0{,}0001$:

- IMBFOA resuelve 18 de las 24 funciones de prueba
- Memetic-SAMSDE resuelve 23 de las 24 funciones de prueba
- APF-GA resuelve 17 de las 24 funciones de prueba
- MDE resuelve 19 de las 24 funciones de prueba

Calidad de los resultados (Con base en la media y al mejor valor encontrado):

- Memetic-SAMSDE tiene mejores resultados en la mayoría de las funciones de prueba excepto g19 y g20
- El segundo algoritmo con mejor calidad de resultados es MDE, seguido de IMBFOA y AP-GA.

Resultados de la prueba de signo de Wilcoxon:

- No hay diferencia significativa entre los resultados de los algoritmos

IMBFOA, en la mayoría de los problemas de prueba, encontró soluciones factibles en todas sus ejecuciones excepto g20, g21 y g22. Sin embargo, la condición de éxito no fue satisfecha en todos las funciones de prueba, en promedio, el 74 % de las 25 ejecuciones realizadas satisfacieron esta condición con un promedio de 198,411 evaluaciones en cada función de prueba. En la Tabla 6.3 se presenta esta información en términos de tasa de factibilidad, tasa de éxito y rendimiento de éxito.

Además, estos resultados se comparan con seis algoritmos del estado del arte, los cuales son: ϵDEag [99], SaDE [114], MPDE [115], GDE [116], JDE [117], y MDE [118]. En la tabla 6.4 se presentan los valores de cada algoritmo en términos de tasa de factibilidad y tasa de éxito. Los mejores algoritmos son ϵDEag, SaDE en términos de tasa de factibilidad y CMODE en tasa de éxito, IMBFOA fue el algoritmo con menos porcentaje en ambas comparaciones.

IMBFOA también se comparó contra los mismos algoritmos de la Tabla 6.4 con el término de rendimiento exitoso (véase la tabla 6.5), donde la propuesta fue mejor en los problemas de prueba g04, g15 y g19 al obtener en menos evaluaciones soluciones factibles que satisfacen la condición de éxito. En la suma de rendimiento exitoso, la propuesta es competitivo en alguno de los problemas de prueba pero es el algoritmo con más funciones de prueba no resueltas. En esta comparación, el mejor algoritmo fue MDE, sin embargo, CMODE es el algoritmo que resuelve más funciones de prueba, 23 de las 24.

Prob.	Criterios	IMBFOA	Memetic-SAMSDE	APF-GA	MDE
	Evaluaciones	240,000	240,000	500,000	500,000
	Ejecuciones	25	30	25	25
	WSRT	=	=	=	=
g01	Mejor	**-15**	**-15**	**-15**	**-15**
-15	Media	-14.93	**-15**	**-15**	**-15**
	Desv.Est.	2.36E-01	**0**	**0**	**0**
g02	Mejor	**-0.8035462**	**-0.8036191**	-0.803601	**-0.8036191**
-0.803619104	Media	-0.6801028	**-0.8036191**	-0.803518	-0.78616
	Desv.Est.	5.98E-02	**0**	1.00E-04	1.20E-02
g03	Mejor	**-1.001**	**-1.0005**	**-1.001**	**-1.005**
-1.0005001	Media	**-1.0009**	**-1.0005**	**-1.001**	**-1.005**
	Desv.Est.	4.57E-05	**0**	**0**	**0**
g04	Mejor	**-30665.539**	**-30665.539**	**-30665.539**	**-30665.5386**
-30665.53867	Media	**-30665.539**	**-30665.539**	**-30665.539**	**-30665.5386**
	Desv.Est.	**0**	**0**	1.00E-04	**0**
g05	Mejor	**5126.497**	**5126.497**	**5126.497**	**5126.497**
5126.496714	Media	5126.496	**5126.497**	5127.5423	**5126.497**
	Desv.Est.	5.18E-05	**0**	1.43E+00	**0**
g06	Mejor	**-6961.813875**	**-6961.813875**	**-6961.814**	**-6961.814**
-6961.813876	Media	-6961.813851	**-6961.813875**	**-6961.814**	**-6961.814**
	Desv.Est.	8.39E-05	**0**	**0**	**0**
g07	Mejor	**24.3062**	**24.3062**	**24.3062**	**24.3062**
24.30620907	Media	24.481	**24.3062**	**24.3062**	**24.3062**
	Desv.Est.	2.62E-01	**0**	**0**	**0**
g08	Mejor	**-0.095825**	**-0.095825**	**-0.095825**	**-0.095825**
-0.095825041	Media	**-0.095825**	**-0.095825**	**-0.095825**	**-0.095825**
	Desv.Est.	8.01E-18	**0**	**0**	**0**
g09	Mejor	**680.63**	**680.63**	**680.63**	**680.63**
680.6300574	Media	680.64	**680.63**	**680.63**	**680.63**
	Desv.Est.	3.70E-02	**0**	**0**	**0**
g10	Mejor	**7049.24802**	**7049.24802**	**7049.24802**	**7049.24802**
7049.248021	Media	7320.5021	**7049.24802**	7077.6821	**7049.24802**
	Desv.Est.	8.10E+02	**0**	5.12E+01	**0**
g11	Mejor	**0.7499**	**0.7499**	**0.7499**	**0.7499**
0.7499	Media	**0.7499**	**0.7499**	**0.7499**	**0.7499**
	Desv.Est.	3.48E-06	**0**	**0**	**0**
g12	Mejor	**-1**	**-1**	**-1**	**-1**
-1	Media	**-1**	**-1**	**-1**	**-1**
	Desv.Est.	**0**	**0**	**0**	**0**
g13	Mejor	**0.053941**	**0.053942**	**0.053942**	**0.053942**
0.053941514	Media	0.17709	**0.053942**	**0.053942**	**0.053942**
	Desv.Est.	1.83E-01	**0**	**0**	**0**
g14	Mejor	-46.467894	**-47.764888**	-47.76479	-47.764887
-47.76488846	Media	-45.016895	**-47.764888**	-47.76479	-47.764874
	Desv.Est.	9.84E-01	**0**	1.00E-04	1.40E-05
g15	Mejor	**961.71502**	**961.71502**	**961.71502**	**961.71502**
961.7150223	Media	**961.71502**	**961.71502**	**961.71502**	**961.71502**
	Desv.Est.	1.10E-08	**0**	**0**	**0**
g16	Mejor	**-1.905155**	**-1.905155**	**-1.905155**	**-1.905155**
-1.905155259	Media	-1.904055	**-1.905155**	**-1.905155**	**-1.905155**
	Desv.Est.	1.25E-03	**0**	**0**	**0**
g17	Mejor	8927.5917	**8853.5397**	8853.5398	**8853.5397**
8853.539675	Media	8927.5918	**8823.5397**	8888.4876	**8853.5397**
	Desv.Est.	1.39E-04	**0**	2.90E+01	**0**
g18	Mejor	**-0.866025**	**-0.866025**	**-0.866025**	**-0.866025**
-0.866025404	Media	-0.864223	**-0.866025**	-0.865925	**-0.866025**
	Desv.Est.	2.46E-03	**0**	1.00E-04	**0**
g19	Mejor	**32.655592**	**32.655593**	**32.655593**	32.64827
32.65559295	Media	37.160608	**32.655593**	**32.655593**	33.34125
	Desv.Est.	8.45E+00	**0**	**0**	8.47E-01
g20	Mejor	-	-	-	-
0.2049794	Media	-	-	-	-
	Desv.Est.	-	-	-	-
g21	Mejor	-	**193.72451**	196.63301	**193.72451**
193.7245101	Media	-	**193.72451**	199.51581	**193.72451**
	Desv.Est.	-	**0**	2.36E+00	**0**
g22	Mejor	-	**236.370313**	-	-
236.4309755	Media	-	**245.738829**	-	-
	Desv.Est.	-	**9.05E+00**	-	-
g23	Mejor	-400.0023	**-400.0551**	-399.7624	**-400.0551**
-400.0551	Media	-399.9196	**-400.0551**	-394.7627	**-400.0551**
	Desv.Est.	2.78E-01	**0**	3.87E+00	**0**
g24	Mejor	**-5.508013**	**-5.508013**	**-5.508013**	**-5.508013**
-5.508013272	Media	**-5.508013**	**-5.508013**	**-5.508013**	**-5.508013**
	Desv.Est.	2.70E-11	**0**	**0**	**0**

Tabla 6.2: Estadística básica y resultados de la prueba de signo de Wilcoxon obtenidos por IMBFOA y otros algoritmos.

En la tabla 6.2 se presenta la medida del Progress ratio (Ver Medidas de rendimiento en el Anexo 8) obtenida por IMBFOA sobre las 24 funciones de prueba. La propuesta tiene una gran capacidad para mejorar la primera solución factible encontrada, sin embargo, en los problemas g05, g07, g09, g10, g13, g15, g17 y g19 la medida es cercana a cero, la mayoría de estos problemas tienen una muy pequeña región factible (ver columna $\vec{\boldsymbol{\rho}}$ de la Tabla 8.1).

Prob.	Tasa de factibilidad	Tasa de éxito	Rendimiento exitoso
g01	100 %	64 %	218,451 FEs
g02	100 %	4 %	3,104,700 FEs
g03	100 %	100 %	26,022 FEs
g04	100 %	100 %	7,707 FEs
g05	100 %	76 %	122,782 FEs
g06	100 %	96 %	49,496 FEs
g07	100 %	44 %	127,975 FEs
g08	100 %	100 %	601 FEs
g09	100 %	76 %	41,549 FEs
g10	100 %	84 %	68,537 FEs
g11	100 %	100 %	78,051 FEs
g12	100 %	100 %	3,991 FEs
g13	100 %	68 %	132,750 FEs
g14	100 %	-	-
g15	100 %	100 %	8,251 FEs
g16	100 %	20 %	588,396 FEs
g17	100 %	-	-
g18	100 %	32 %	85,422 FEs
g19	100 %	68 %	82,566 FEs
g20	-	-	-
g21	-	-	-
g22	-	-	-
g23	100 %	-	-
g24	100 %	100 %	4,089 FEs
Media	87.5 %	74 %	198,411 FEs

Tabla 6.3: Tasa de factibilidad, tasa de éxito y Rendimiento exitoso de IMBFOA

	Tasa de factibilidad								Tasa de éxito							
Prob.	ϵDEag	SaDE	MPDE	GDE	JDE-2	MDE	CMODE	IMBFOA	ϵDEag	SaDE	MPDE	GDE	JDE-2	MDE	CMODE	IMBFOA
g02	100 %	100 %	100 %	100 %	100 %	100 %	100 %	100 %	100 %	84 %	92 %	72 %	92 %	16 %	100 %	4 %
g03	100 %	100 %	100 %	96 %	100 %	100 %	100 %	100 %	100 %	96 %	84 %	4 %	-	100 %	100 %	100 %
g05	100 %	100 %	100 %	96 %	100 %	100 %	100 %	100 %	100 %	100 %	100 %	92 %	68 %	100 %	100 %	76 %
g11	100 %	100 %	100 %	100 %	100 %	100 %	100 %	100 %	100 %	100 %	96 %	100 %	96 %	100 %	100 %	100 %
g13	100 %	100 %	88 %	88 %	100 %	100 %	100 %	100 %	100 %	100 %	48 %	40 %	0 %	100 %	100 %	68 %
g14	100 %	100 %	100 %	100 %	100 %	100 %	100 %	100 %	100 %	80 %	100 %	96 %	100 %	100 %	100 %	-
g15	100 %	100 %	100 %	100 %	100 %	100 %	100 %	100 %	100 %	100 %	100 %	96 %	96 %	100 %	100 %	100 %
g17	100 %	100 %	96 %	76 %	100 %	100 %	100 %	100 %	100 %	4 %	28 %	16 %	4 %	100 %	100 %	-
g18	100 %	100 %	100 %	84 %	100 %	100 %	100 %	100 %	100 %	92 %	100 %	76 %	100 %	100 %	100 %	32 %
g19	100 %	100 %	100 %	100 %	100 %	100 %	100 %	100 %	100 %	100 %	100 %	88 %	100 %	-	100 %	68 %
g20	-	-	-	-	4 %	-	-	-	-	-	-	-	-	-	100 %	-
g21	100 %	100 %	100 %	88 %	100 %	100 %	100 %	-	100 %	60 %	68 %	60 %	92 %	100 %	80 %	-
g22	100 %	100 %	-	-	-	-	-	-	-	-	-	-	-	-	-	-
g23	100 %	100 %	100 %	88 %	100 %	100 %	100 %	100 %	100 %	88 %	100 %	40 %	92 %	100 %	100 %	-
Average	**95.83** %	**95.83** %	91 %	88.17 %	91.83 %	91.67 %	91.67 %	87.5 %	91.67 %	83.5 %	84 %	74.17 %	76.67 %	84 %	**95** %	74 %

Tabla 6.4: Resultados de IMBFOA en términos de tasa de factibilidad y tasa de éxito comparado con otros algoritmos.

	Rendimiento exitoso						
Prob.	ϵDEag	MPDE	GDE	JDE-2	MDE	CMODE	IMBFOA
g01	5.90E+04	4.30E+04	**4.10E+04**	5.00E+04	7.50E+04	1.20E+05	2.18E+05
g02	1.50E+05	3.00E+05	1.50E+05	1.50E+05	**6.00E+04**	1.90E+05	3.10E+06
g03	8.90E+04	**2.50E+04**	3.50E+06	-	4.50E+04	7.50E+04	2.60E+04
g04	2.60E+04	2.10E+04	1.50E+04	4.10E+04	4.20E+04	7.30E+04	**7.70E+03**
g05	9.70E+04	2.20E+05	1.90E+05	4.50E+05	**2.10E+04**	2.90E+04	1.23E+05
g06	7.40E+03	1.10E+04	6.50E+03	2.90E+04	**5.20E+03**	3.50E+04	4.95E+04
g07	7.40E+04	**5.70E+04**	1.20E+05	1.30E+05	1.90E+05	1.60E+05	1.28E+05
g08	1.10E+04	1.50E+03	1.50E+03	3.20E+03	**9.20E+02**	5.90E+03	6.01E+02
g09	2.30E+04	2.10E+04	3.00E+04	5.50E+04	**1.60E+04**	7.10E+04	4.15E+04
g10	1.10E+05	**4.80E+04**	8.30E+04	1.50E+05	1.60E+05	1.80E+05	6.85E+04
g11	1.60E+04	2.30E+04	8.50E+03	5.40E+04	**3.00E+03**	6.00E+03	7.81E+04
g12	4.10E+03	4.20E+03	3.10E+03	6.40E+03	**1.30E+03**	5.00E+03	3.99E+03
g13	3.50E+04	7.40E+05	8.70E+05	-	**2.20E+04**	3.10E+04	1.33E+05
g14	1.10E+05	**4.30E+04**	2.30E+05	9.80E+04	2.90E+05	1.10E+05	-
g15	8.40E+04	2.00E+05	7.50E+04	2.40E+05	1.00E+04	1.30E+04	**8.25E+03**
g16	1.30E+04	1.30E+04	1.30E+04	3.20E+04	**8.70E+03**	2.90E+04	5.88E+05
g17	9.90E+04	7.30E+05	2.10E+06	1.10E+07	**2.60E+04**	1.40E+05	-
g18	5.90E+04	**4.40E+04**	4.80E+05	1.00E+05	1.10E+05	1.10E+05	8.54E+04
g19	3.50E+05	**1.20E+05**	2.30E+05	2.00E+05	-	2.50E+05	-
g20	-	-	-	-	-	4.40E+05	**1.05E+04**
g21	1.40E+05	2.10E+05	5.80E+05	1.30E+05	**1.10E+05**	1.30E+05	-
g22	-	-	-	-	-	-	-
g23	**2.00E+05**	2.10E+05	1.10E+06	3.60E+05	3.60E+05	2.40E+05	-
g24	3.00E+03	4.30E+03	3.10E+03	1.00E+04	**1.80E+03**	2.20E+04	4.09E+03

Suma de rendimiento exitoso	
ϵDEag	1.70E+06+2*-
MPDE	3.10E+06+2*-
GDE	9.80E+06+2*-
jDE-2	1.30E+07+4*-
MDE	1.50E+06+3*-
CMODE	**2.50E+06+1***-
MBFOAsqp	4.76E+06+6*-

Tabla 6.5: Resultados de IMBFOA en términos de rendimiento exitoso comparado con otros algoritmos.

IMBFOA en problemas de prueba del CEC2010 con 10 dimensiones

IMBFOA fue probada en las funciones de prueba del CEC2010 con 10 dimensiones, en la Tabla 6.7 son presentados los resultados y se comparan contra Memetic-SAMSDE, **ϵ**DEag (el algoritmo

Prob.	Mejor	Media	Desv.Est.
g01	1.33E+00	1.42E+00	5.26E-02
g02	1.57E+00	1.57E+00	5.87E-03
g03	1.57E+00	1.62E+00	1.41E-01
g04	5.17E+00	5.17E+00	1.14E-03
g05	1.15E-07	1.30E-07	5.05E-09
g06	4.42E+00	4.46E+00	6.79E-02
g07	4.32E-07	7.06E-01	5.14E-01
g08	1.60E+00	1.60E+00	6.97E-04
g09	2.22E-16	1.23E-01	1.05E-01
g10	6.00E-15	7.39E-02	1.03E-01
g11	1.44E-01	1.44E-01	2.32E-06
g12	1.57E+00	1.57E+00	5.16E-04
g13	5.53E-05	5.10E-01	5.70E-01
g14	1.86E+00	1.90E+00	2.30E-02
g15	7.79E-08	7.79E-08	5.72E-12
g16	3.69E-01	1.05E+00	6.09E-01
g17	2.96E-07	3.29E-07	7.76E-09
g18	1.57E+00	1.68E+00	4.09E-02
g19	4.44E-16	6.11E-01	8.08E-01
g20	-	-	-
g21	-	-	-
g22	-	-	-
g23	2.99E+00	2.99E+00	6.98E-04
g24	6.28E-01	6.35E-01	2.21E-02

Tabla 6.6: Estadística básica de los resultados de la medida Progress ratio en IMBFOA sobre el benchmark del CEC2006.

ganador de la competencia del CEC2010) y IEMA. Para comparar los resultados y saber si se cumple la condición de éxito, se tomará como mejor valor conocido $\boldsymbol{f(\overrightarrow{x^*})}$ para cada función los resultados obtenidos por $\boldsymbol{\epsilon}$DEag, el algoritmo ganador en CEC 2010.

Con base en los resultados obtenidos por la propuesta se obtiene el siguiente análisis:

Funciones de prueba que resuelve al satisfacer la condición $\boldsymbol{f(\overrightarrow{x}) - f(\overrightarrow{x^*}) \leq 0{,}0001}$:

- IMBFOA resuelve 12 de las 18 funciones de prueba
- Memetic-SAMSDE resuelve 18 de las 18 funciones de prueba
- IEMA resuelve 16 de las 18 funciones de prueba

Calidad de resultados (Con base en la media y al mejor valor encontrado):

- Memetic-SAMSDE tiene los mejores resultados seguido de $\boldsymbol{\epsilon}$DEag, IMBFOA y por último, IEMA.

Funciones de prueba donde se encuentran soluciones factibles:

- IMBFOA encuentra soluciones factibles a la mayoría de las funciones de prueba excepto $\boldsymbol{C11}$
- Memetic-SAMSDE y IEMA encuentran soluciones factibles a todas las funciones de prueba

Resultados de la prueba de signo de Wilcoxon:

- No existe una diferencia significativa entre los resultados de los algoritmos IMBFOA, $\boldsymbol{\epsilon}$DEag y IEMA con respecto al mejor valor encontrado, pero entre IMBFOA y Memetic-SAMSDE si hay una diferencia significativa a favor de Memetic-SAMSDE.

Prob.	Criterios	IMBFOA	Memetic-SAMSDE	ϵDEag	IEMA
	FEs	200,000	200,000	200,000	200,000
	Ejecuciones	25	25	25	25
	WSRT		+	=	=
C01	Mejor	**-0.7473104**	**-0.7473104**	**-0.7473104**	**-0.74731**
	Media	-0.723775	**-0.7473104**	-0.7470402	-0.743189
	Desv.Est.	3.14E-02	**0**	1.32E-03	4.33E-03
C02	Mejor	**-2.2777066**	**-2.2777099**	**-2.277702**	**-2.27771**
	Media	-2.0803575	-2.2776477	-2.25887	**-2.27771**
	Desv.Est.	3.64E-01	5.9704E-05	2.39E-02	**1.82E-07**
C03	Mejor	7.20E-05	**0**	**0**	1.47E-16
	Media	1.242E+11	4.8159E-21	**0**	6.23E-07
	Desv.Est.	4.65E+11	6.862E-21	**0**	1.40E-06
C04	Mejor	**-6.04E-04**	-1.00E-05	-9.99E-06	-9.99E-06
	Media	1.02E-01	**-1.00E-05**	-9.92E-06	-9.91E-06
	Desv.Est.	2.76E-01	**2.28E-10**	1.55E-07	8.99E-08
C05	Mejor	-483.599468	-483.610625	-483.6106	**-483.611**
	Media	-296.059023	**-483.610625**	-483.6106	-379.156
	Desv.Est.	1.60E+02	1.4883E-07	**3.89E-13**	1.79E+02
C06	Mejor	-578.66231	**-578.66236**	-578.6581	-578.662
	Media	-507.301756	**-578.6622**	-578.6528	-551.47
	Desv.Est.	1.13E+02	**1.18E-04**	3.63E-003	7.36E+01
C07	Mejor	7.13E-09	**0**	**0**	1.75E-08
	Media	9.57E-01	9.30E-24	**0**	3.26E-09
	Desv.Est.	1.74E+00	1.71E-23	**0**	3.39E-09
C08	Mejor	2.95E-09	**0**	**0**	1.01E-10
	Media	7.37E+01	**9.51E-20**	6.73E+00	4.07E+00
	Desv.Est.	1.55E+02	**3.49E-19**	5.56E+00	6.38E+00
C09	Mejor	2.81E-11	**0**	**0**	1.20E-09
	Media	3.28E+07	1.29E-21	**0**	1.95E+12
	Desv.Est.	1.33E+08	2.87E-21	**0**	5.40E+12
C10	Mejor	4.08E+01	**0**	**0**	5.40E-09
	Media	4.29E+06	7.46E-23	**0**	2.56E+12
	Desv.Est.	7.08E+06	1.56E-22	**0**	3.97E+12
C11	Mejor	-	**1.52E-03**	**1.52E-03**	**1.52E-03**
	Media	-	**1.52E-03**	**1.52E-03**	-1.15E-03
	Desv.Est.	-	1.36E-08	**6.34E-11**	2.73E-08
C12	Mejor	-0.1992	**-570.0899**	**-570.0899**	-10.9735
	Media	-0.1948	-3.3553	**-336.7349**	-0.648172
	Desv.Est.	**1.14E-02**	1.16E+02	1.78E+02	2.20E+00
C13	Mejor	-62.27639	**-68.42937**	**-68.42937**	**-68.4294**
	Media	-58.15052	**-68.42937**	**-67.42937**	-68.0182
	Desv.Est.	2.63E+00	**0**	1.03E-06	1.40E+00
C14	Mejor	4.08E-08	**0**	**0**	8.04E-10
	Media	2.22E+05	9.78E-21	**0**	5.63E+01
	Desv.Est.	8.34E+05	2.10E-20	**0**	1.83E+02
C15	Mejor	4.50E+00	**0**	**0**	9.35E-10
	Media	2.22E+07	**2.48E-20**	1.80E-01	1.58E+08
	Desv.Est.	5.45E+07	**1.01E-19**	8.81E-01	6.04E+08
C16	Mejor	**0**	**0**	**0**	4.44E-16
	Media	3.81E-02	**0**	3.70E-01	3.30E-02
	Desv.Est.	7.99E-02	**0**	3.71E-01	2.26E-02
C17	Mejor	3.07E-16	**0**	1.46E-017	9.48E-15
	Media	1.00E+00	**8.17E-16**	1.25E-01	3.15E-03
	Desv.Est.	1.50E+00	**1.45E-15**	1.94E-01	1.58E-02
C18	Mejor	5.61E-16	**0**	3.73E-20	2.24E-15
	Media	4.09E-10	**3.04E-25**	9.68E-19	1.62E-14
	Desv.Est.	1.31E-09	**1.50E-24**	1.81E-18	3.82E-14

Tabla 6.7: Estadísticas básicas y resultado de la prueba de signo de Wilcoxon obtenidos por IMBFOA comparados con otros algoritmos en el benchmark CEC2010 con 10 dimensiones.

La propuesta en este benchmark encontró soluciones factibles a la mayoría de los problemas, excepto en la función $\boldsymbol{C11}$ (ver Tabla 6.8). En las funciones $\boldsymbol{C12}$ y $\boldsymbol{C04}$ el algoritmo solo obtuvo el 76 % y 92 % de tasa de factibilidad, respectivamente. En términos de tasa de éxito la propuesta no obtuvo soluciones factibles cercanas a la mejor solución conocida en las funciones C05, C10, C12 y C15. Sin embargo, en la función C04 la propuesta obtuvo el mejor valor de aptitud en comparación con el resto de los algoritmos con los que se comparó. En promedio la propuesta necesita 318,812 evaluaciones para encontrar una solución factible similar a la mejor conocida.

La medida Progress ratio en este benchmarck por IMBFOA es buena, si observamos la media en la Tabla 6.9 solo en las funciones C03, C09, C10 y C15 los resultados son cercanos a cero, en las demás,

Prob.	Tasa de factibilidad	Tasa de éxito	Rendimiento exitoso
C01	100 %	16 %	687,359 FEs
C02	100 %	28 %	269,591 FEs
C03	100 %	4 %	3,757,000 FEs
C04	92 %	80 %	33,049 FEs
C05	100 %	-	-
C06	100 %	24 %	280,688 FEs
C07	100 %	76 %	39,556 FEs
C08	100 %	32 %	171,753 FEs
C09	100 %	8 %	131,325 FEs
C10	100 %	-	-
C11	-	-	-
C12	76 %	-	-
C13	100 %	-	-
C14	100 %	52 %	195,961 FEs
C15	100 %	-	-
C16	100 %	68 %	84,313 FEs
C17	100 %	32 %	77,255 FEs
C18	100 %	100 %	10,769 FEs
Media	98.12 %	43.33 %	318,812 FEs

Tabla 6.8: Tasa de factibilidad, tasa de éxito y rendimiento exitoso de IMBFOA sobre el benchmark del CEC2010 con 10 dimensiones.

Prob.	Mejor	Media	Desv.Est.
C01	1.57E+00	1.57E+00	1.95E-03
C02	5.74E-01	1.03E+00	5.48E-01
C03	2.82E-04	9.26E-01	2.49E+00
C04	2.64E-03	1.40E+00	5.53E-01
C05	2.23E+00	3.08E+00	3.90E-01
C06	2.82E+00	3.20E+00	2.08E-01
C07	9.67E-11	2.39E+00	5.17E+00
C08	5.37E-14	2.41E+00	4.01E+00
C09	8.82E-09	6.13E-01	1.66E+00
C10	1.45E-08	6.63E-06	1.40E-05
C11	-	-	-
C12	2.38E-01	1.51E+00	3.10E-01
C13	2.00E+00	2.10E+00	5.54E-02
C14	2.30E-10	9.78E+00	6.53E+00
C15	4.44E-16	1.33E-04	6.00E-04
C16	2.22E-16	5.68E+00	6.78E+00
C17	3.11E-15	1.76E+00	3.54E+00
C18	6.01E-01	2.98E+00	3.13E+00

Tabla 6.9: Estadística básica de los resultados de la medida Progress ratio en IMBFOA sobre el benchmark del CEC2010 con 10 dimensiones.

la propuesta mejora la primera solución factible encontrada considerablemente.

IMBFOA en funciones de prueba del CEC2010 con 30 dimensiones

Los resultados obtenidos por IMBFOA en las funciones de prueba del benchmark del CEC2010 con 30 dimensiones son presentados en la Tabla 6.10, y son comparados contra Memetic-SAMSDE, ϵDEag y IEMA. Con base en los resultados obtenidos por la propuesta se obtiene el siguiente análisis tomando como referencia a ϵDEag como el algoritmo con el mejor valor conocido $\boldsymbol{f(\overrightarrow{x^*})}$ para cada función:
Funciones de prueba que resuelve al satisfacer la condición $\boldsymbol{f(\overrightarrow{x}) - f(\overrightarrow{x^*}) \leq 0{,}0001}$:

- IMBFOA resuelve 11 de las 18 funciones de prueba
- Memetic-SAMSDE resuelve 18 de las 18 funciones de prueba
- IEMA resuelve 11 de las 18 funciones de prueba

Calidad de resultados (Con base en la media y al mejor valor encontrado):

- Memetic-SAMSDE tiene los mejores resultados seguido de ϵDEag, IMBFOA y por último, IEMA

Funciones de prueba donde se encuentran soluciones factibles:

- IMBFOA encuentra soluciones factibles a la mayoría de las funciones de prueba excepto C03
- Memetic-SAMSDE encuentra soluciones factibles a todas las funciones de prueba
- IEMA encuentra soluciones factibles a la mayoría de las funciones de prueba excepto C03, C04, C11 y C12

Resultados de la prueba de signo de Wilcoxon:

- No existe una diferencia significativa entre los algoritmos IMBFOA y IEMA con respecto al mejor valor de aptitud encontrado
- Existe una diferencia significativa a favor de MBFOAsqp cuando es comparado contra ϵDEag
- Existe una diferencia significativa a favor de Memetic-SAMSDE al compararlo contra IMBFOA

En los resultados de la Tabla 6.10 se puede observar que IMBFOA obtiene el mejor valor de aptitup para las funciones $\boldsymbol{C04}$ y $\boldsymbol{C17}$ en comparación de los demás algoritmos.

IMBFOA en el benchmark CEC2010 con 30 dimensiones tiene un comportamiento similar que con 10 dimensiones en términos de tasa de factibilidad, tasa de éxito y rendimiento éxitoso (ver Figura 6.11). A diferencia de los resultados obtenidos con 10 dimensiones, la propuesta no puede encontrar ninguna solución factible a la función $\boldsymbol{C03}$, sin embargo, para $\boldsymbol{C11}$ la propuesta obtiene soluciones factibles que satisfacen la condición de éxito. El porcentaje del promedio de la tasa de éxito disminuye y el número de evaluaciones promedio para encontrar una solución similar o mejor que la solución mejor conocida incrementa debido a la complejidad de las funciones y al número de variables de decisión.

En la Tabla 6.12 la medida Progress ratio para el benchmark CEC2010 con 30 dimensiones es presentado, en las funciones C02, C04, C07, C09 y C14 los resultados son cercanos a cero. En el resto de las funciones, la propuesta mejora considerablemente la primera solución factible encontrada.

Para futuras referencias, en la Tabla 6.13 es presentado el mejor valor de aptitud para cada función de prueba del benchmark CEC2010 con 10 y 30 dimensiones. Este es el mejor valor de todos los algoritmos usados en las tablas de comparación de este trabajo (IMBFOA, Memetic-SAMDES, ϵDEag, y IEMA) que resuelven este benchmark.

Prob.	Criterios	IMBFOA	Memetic-SAMSDE	ϵDEag	IEMA
	FEs	600,000	600,000	600,000	600,000
	Ejecuciones	25	25	25	25
	WSRT		+		=
C01	Mejor	-0.3996041	**-0.8218843**	-0.8218255	-0.821883
	Media	-0.2914506	-0.8156324	**-0.8208687**	-0.817769
	Desv.Est.	3.79E-02	4.41E-03	**7.10E-04**	4.79E-03
C02	Mejor	-2.2806521	**-2.2880962**	-2.169248	-2.28091
	Media	-2.2191412	**-2.2777017**	-2.151424	-1.50449
	Desv.Est.	1.90E-01	**9.85E-04**	1.20E-02	2.14E+00
C03	Mejor	-	**1.15E-20**	2.87E+01	-
	Media	-	**9.85E-18**	2.88E+01	-
	Desv.Est.	-	**3.48E-17**	8.05E-01	-
C04	Mejor	**3.50E-08**	-3.33E-06	4.70E-03	-
	Media	1.44E-04	**1.51E-05**	8.16E-03	-
	Desv.Est.	2.16E-04	**9.11E-05**	3.07E-03	-
C05	Mejor	-483.59086	**-483.61062**	-453.1307	-286.678
	Media	-181.502159	**-483.61058**	-449.546	-270.93
	Desv.Est.	2.76E+02	**9.60E-05**	2.90E+00	1.41E+01
C06	Mejor	-530.543	**-530.637**	-528.575	-529.593
	Media	-527.2404	**-530.098**	-527.9068	-132.876
	Desv.Est.	6.55E+00	3.08E-01	**4.75E-01**	5.61E+02
C07	Mejor	3.47E-10	**5.02E-25**	1.15E-15	4.82E-10
	Media	1.59E+00	**9.34E-20**	2.60E-15	8.49E-10
	Desv.Est.	1.99E+00	**1.67E-19**	1.23E-15	4.84E-10
C08	Mejor	2.96E-10	**3.40E-21**	2.52E-14	1.12E-09
	Media	1.13E+01	**1.65E-17**	7.83E-14	1.77E+01
	Desv.Est.	2.94E+01	**3.89E-17**	4.86E-14	4.08E+01
C09	Mejor	3.92E-11	**9.82E-23**	2.77E-16	7.31E+03
	Media	4.10E+06	**1.38E-14**	1.07E+01	2.99E+07
	Desv.Est.	6.22E+06	**4.59E-14**	2.82E+01	4.50E+07
C10	Mejor	3.67E-11	**4.88E-25**	3.25E+01	2.77E+04
	Media	3.21E+03	**1.62E-15**	3.33E+01	1.58E+07
	Desv.Est.	5.49E+03	**3.64E-15**	4.55E-01	1.68E+07
C11	Mejor	-3.33E-04	**-3.92E-04**	-3.27E-04	-
	Media	-1.11E-04	**-3.92E-04**	-2.86E-04	-
	Desv.Est.	9.04E-05	**4.80E-07**	2.71E-05	-
C12	Mejor	-0.1992434	**-0.1992611**	-0.1991453	-
	Media	1.38E+00	**-1.99E-01**	3.56E+02	-
	Desv.Est.	6.33E+00	**2.01E-06**	2.89E+02	-
C13	Mejor	-62.7518	**-68.42936**	-66.42473	**-68.4294**
	Media	-59.2295	**-68.2398**	-65.3531	-67.4872
	Desv.Est.	2.20E+00	3.45E-01	5.73E-01	**9.84E-01**
C14	Mejor	1.26E-08	**1.57E-19**	5.02E-14	3.29E-09
	Media	2.28E+06	2.81E-12	**3.09E-13**	7.38E-09
	Desv.Est.	6.28E+06	6.25E-12	**5.61E-13**	3.07E-01
C15	Mejor	1.06E-08	**7.47E-21**	2.16E+01	3.12E+04
	Media	2.69E+04	**3.94E-15**	2.16E+01	2.29E+08
	Desv.Est.	8.79E+04	**1.31E-14**	1.10E-04	4.64E+08
C16	Mejor	1.44E-15	**0**	0	6.16E-12
	Media	5.16E-01	**0**	2.17E-21	1.63E-03
	Desv.Est.	5.01E-01	**0**	1.06E-20	8.16E-03
C17	Mejor	**3.47E-15**	6.91E-11	2.17E-01	9.28E-10
	Media	6.54E+01	**1.23E-07**	6.33E+00	8.84E-02
	Desv.Est.	2.60E+02	**1.48E-07**	4.99E+00	1.51E-01
C18	Mejor	2.78E-15	**3.00E-20**	1.23E+00	1.38E-14
	Media	4.58E-11	**8.36E-15**	8.75E+01	4.74E-14
	Desv.Est.	8.32E-11	3.84E-14	1.66E+02	**6.57E-14**

Tabla 6.10: Estadísticas básicas y resultados de la prueba de signo de Wilcoxon (WSRT) obtenidos por IMBFOA comparados con otros algoritmos en el benchmark del CEC2010 con 30 dimensiones.

En general, IMBFOA:

- Obtiene resultados competitivos a otros algoritmos del estado del arte.
- En problemas altamente restringidos y de alta dimensionalidad carece de robuztes
- La presencia de un buscador local beneficia a MBFOA, sin embargo, el algoritmo es capaz de generar soluciones factibles desde las primeras generaciones, por lo tanto, SQP solo es utilizado dos veces en las iteraciones del algoritmo.
- Los operadores de nados propuestos mejoran la explotación y exploración del espacio de búsqueda pero es necesario hacer un estudio más a fondo o incluir nuevos operadores de nado que

Prob.	Tasa de factibilidad	Tasa de éxito	Rendimiento exitoso
C01	100 %	-	-
C02	100 %	-	-
C03	-	-	-
C04	68 %	44 %	965,601 FEs
C05	100 %	-	-
C06	100 %	-	-
C07	100 %	60 %	63,231 FEs
C08	100 %	80 %	173,904 FEs
C09	100 %	12 %	420,908 FEs
C10	100 %	-	-
C11	32 %	48 %	10,592,075 FEs
C12	92 %	24 %	2,139,503 FEs
C13	100 %	-	-
C14	100 %	4 %	760,875 FEs
C15	100 %	28 %	916,357 FEs
C16	100 %	12 %	2,797,425 FEs
C17	100 %	16 %	881,653 FEs
C18	100 %	100 %	50,782 FEs
Media	93.65 %	38.91 %	1,699,489 FEs

Tabla 6.11: Tasa de factibilidad, tasa de éxito y rendimiento exitoso de IMBFOA sobre el benchmark del CEC2010 con 30 dimensiones.

Prob.	Mejor	Media	Desv.Est
C01	1.58E+00	1.59E+00	4.68E-03
C02	6.35E-01	7.63E-01	3.12E-01
C03	-	-	-
C04	3.85E-01	8.50E-01	3.08E-01
C05	3.62E-08	2.50E+00	1.15E+00
C06	3.13E+00	3.16E+00	6.32E-02
C07	6.77E-11	7.10E-01	2.27E+00
C08	2.67E-01	3.90E+00	5.34E+00
C09	5.30E-09	3.61E-01	1.06E+00
C10	2.22E-07	1.04E+00	2.98E+00
C11	1.61E+00	1.61E+00	9.79E-06
C12	1.18E-03	1.14E+00	7.41E-01
C13	2.03E+00	2.09E+00	3.39E-02
C14	8.88E-16	5.69E-01	8.50E-01
C15	4.66E-15	2.44E+00	4.16E+00
C16	8.39E-06	1.33E+00	3.53E+00
C17	4.66E-07	2.56E+00	3.57E+00
C18	3.31E-06	2.26E+00	1.99E+00

Tabla 6.12: Estadísticas de los resultados de la medida Progress ratio por IMBFOA sobre el benchmark CEC2010 con 30 dimensiones.

perimitan una mejora significativa al proceso de búsqueda.

- La convergencia prematura es el principal problema del algoritmo propuesto.

Prob.	**Mejor valor de aptitud**	
	10d	**30d**
C01	-0.7473104	-0.8218843
C02	-2.27771	-2.2880962
C03	0	1.15E-20
C04	-6.04E-04	3.50E-08
C05	483.611	-483.61062
C06	-578.66236	-530.637
C07	0	5.02E-25
C08	0	3.40E-21
C09	0	9.82E-23
C10	0	4.88E-25
C11	1.52E-03	-3.92E-04
C12	-570.0899	-0.1992611
C13	-68.42937	-68.42936
C14	0	1.57E-19
C15	0	7.47E-21
C16	0	0
C17	0	3.47E-15
C18	0	3.00E-20

Tabla 6.13: Mejor valor de aptitup para las funciones del benchmark CEC2010 con 10 y 30 dimensiones

Capítulo 7

TS-MBFOA

En este capítulo se describe a la propuesta llamada TS-MBFOA (por sus siglas en Inglés Two Swim-Modified Bacterial Foraging Optimization Algorithm) la cual intercala dos nados en el proceso quimiotáxico, el primero es el nado original con tamaño de paso aleatorio y el segundo nado incluye el operador de mutación usado en los algoritmos evolutivos para mejorar la capacidad de exploración y explotación del algoritmo. Aunque en el estado del arte existen propuestas basados en BFOA que utilizan el operador de mutación dentro del proceso de búsqueda [33, 53, 58, 76, 79, 86, 85], ninguna de ellas incluye el operador de mutación como mecanismo de nado.

7.1. Mecanismos de TSMBFOA

En esta sección se describe el mecanismo propuesto para mejorar el rendimiento del algoritmo IMBFOA y el pseudocódigo derivado.

7.1.1. Operadores de nado

En proceso de quimiotáxis dos nados se intercalan, en cada ciclo solo un nado de explotación o exploración es realizado. El proceso comienza con el nado de explotación (nado clásico). Sin embargo, una bacteria no necesariamente intercalará exploración y explotación en los nados, ya que si la nueva posición de un nado dado, $\boldsymbol{\theta^i(j+1,G)}$ tiene una mejor aptitud (basado en las reglas de factibilidad) que la posición original $\boldsymbol{\theta^i(j,G)}$, otro nado en la misma dirección se llevará a cabo en el siguiente ciclo. De lo contrario, un nuevo giro será calculado. El proceso se detiene después de $\boldsymbol{N_c}$ intentos.

El nado de exploración usa la mutación entre bacterias y es calculado con la Ecuación 7.1:

$$\boldsymbol{\theta^i(j+1,G) = \theta^i(j,G) + (\beta - 1)(\theta^r_1(j,G) - \theta^r_2(j,G))} \tag{7.1}$$

donde $\boldsymbol{\theta^r_1(j,G)}$ y $\boldsymbol{\theta^r_2(j,G)}$ son dos bacterias diferentes seleccionados aleatoriamente de la población. $\boldsymbol{\beta}$ es un parámetro definido por el usuario utilizado en el operador de agrupamiento el cual define la cercanía de la nueva posición de una bacteria con respecto a la posición de la mejor bacteria de la población, en este operador, $\boldsymbol{\beta - 1}$ es un parámetro de control positivo para escalar los diferentes vectores en (0,1], es decir, escalas de la zona donde una bacteria puede moverse.

La figura 7.1 muestra el comportamiento de este operador de nado utilizando un espacio de dos variables de decisión, cada uno en dentro del rango [-5, 5].
El nado de explotación es calculado con el Ecuación 7.2:

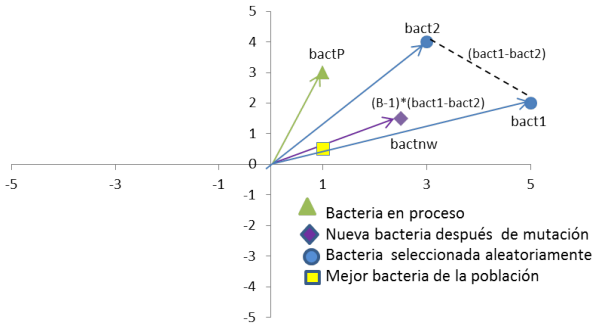

Figura 7.1: Comportamiento del nado de exploración.

$$\boldsymbol{\theta^i(j+1,G) = \theta^i(j,G) + C(i,G)\phi(i)} \tag{7.2}$$

donde $\boldsymbol{\phi(i)}$ se calcula con el operador de giro original de BFOA (Ecuación 3.6) y $\boldsymbol{C(i,G)}$ es el tamaño de paso aleatorio de cada bacteria actualizado con la Ecuación 7.3:

$$\boldsymbol{C(i,G) = R * \Theta(i)} \tag{7.3}$$

donde $\boldsymbol{\Theta(i)}$ es un vector generado de forma aleatoria de tamaño $\boldsymbol{n}$ con elementos dentro del rango de cada variable de decisión decisión: $\boldsymbol{[U_k, L_k]}$, $\boldsymbol{k = 1, ...n}$, y $\boldsymbol{R}$ es un parámetro definido por el usuario para escalar el tamaño de paso, este valor debe ser cercana a cero, por ejemplo. *5.00e-04*. La inicial $\boldsymbol{C(i,0)}$ se genera utilizando $\boldsymbol{\theta(i)}$. Este tamaño de paso aleatorio permite que las bacterias se puedan mover en diferentes direcciones dentro del espacio de búsqueda y evita la convergencia prematura como se sugiere en [35].

La figura 7.2 muestra el comportamiento del nado de explotación, este ejemplo utiliza el mismo espacio de búsqueda de la figura 7.1. En este caso, la bacteria en proceso buscará en un espacio más pequeño cerca de su punto original, la zona marcada con el círculo dibujado con líneas de puntos.

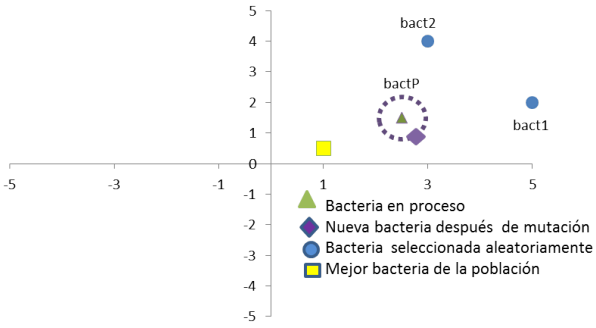

Figura 7.2: Comportamiento del nado de explotación.

Es importante remarcar que el nado de exploración (Ecuación 7.1) realiza movimientos más grandes debido a la mutación con bacterias seleccionadas aleatoriamente dentro del espacio de búsqueda. Por otro lado, el nado de explotación (Ecuación 7.2) genera pequeños movimientos utilizando el tamaño de paso aleatorio en el proceso de búsqueda.

En el ciclo medio del proceso quimiotáxico es aplicado el operador de agrupamiento con la Ecuación 3.4, un ejemplo de su comportamiento se muestra en la figura 7.3, donde $\boldsymbol{\beta}$ es un parámetro

positivo definido por el usuario entre (1, 2]. Sin embargo, en esta propuesta si una solución viola el límite de las variables de decisión, a diferencia de MBFOA, una nueva solución de $\boldsymbol{x_i}$ es generada aleatoriamente entre los límites inferior y superior $\boldsymbol{L_i} \leq \boldsymbol{x_i} \leq \boldsymbol{U_i}$ de las variables de decisión.

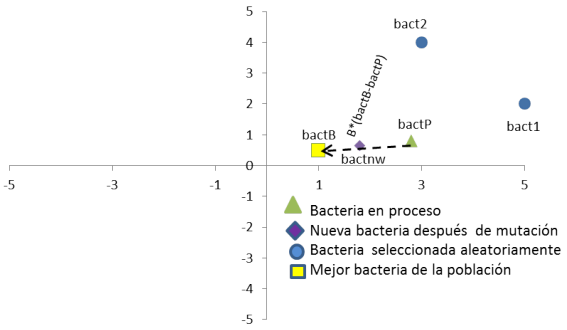

Figura 7.3: Comportamiento de agrupamiento.

La figura 7.4 muestra la posición final de la bacteria después de un nado de exploración, un nado de explotación y un agrupamiento. Sin embargo, este comportamiento puede ser diferente debido a que los nados no son secuenciales en el proceso quimiotáxico.

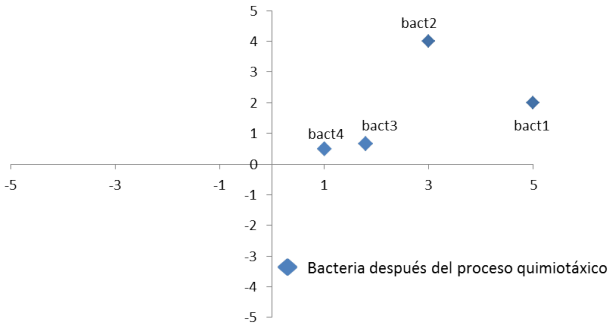

Figura 7.4: Bacterias después de un nado de exploración, un nado de explotación y un agrupamiento.

Es importante mencionar que el operador de búsqueda local SQP es utilizado solo en la primera y en el ciclo medio de las generaciones del algoritmo. La estructura de TS-MBFOA se muestra en la figura 7.5, el pseudocodigo correspondiente se presenta en el algoritmo 3 donde los parámetros de entrada son: número de bacterias $\boldsymbol{S_b}$, límite del ciclo quimiotáxico $\boldsymbol{N_c}$, número de bacterias a reproducir $\boldsymbol{S_r}$, factor de escalamiento $\boldsymbol{\beta}$ para el operador de agrupamiento, $\boldsymbol{R}$ factor de escalamiento para el tamaño de paso, frecuencia de la reproducción $\boldsymbol{RepCycle}$, número de generaciones $\boldsymbol{GMAX}$, frecuencia del buscador local $\boldsymbol{LS_G}$. Las modificaciones y nuevos mecanismos propuestos son remarcados con letras negritas.

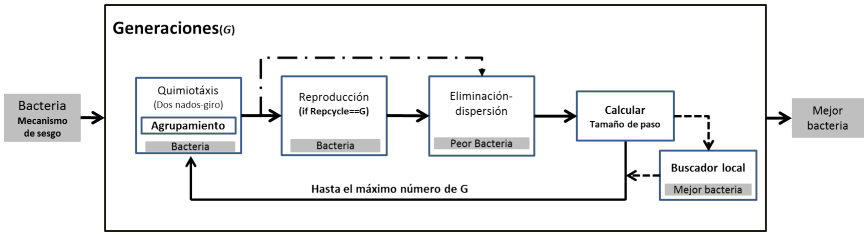

Figura 7.5: Procesos generales de TS-MBFOA.

Crear una población inicial de bacterias usando el mecanismo de sesgo $\boldsymbol{\theta^i(j,0)}$ $\forall \boldsymbol{i}$, $\boldsymbol{i = 1, \ldots, S_b}$
Evaluar $\boldsymbol{f(\theta^i(j,0))}$ $\forall \boldsymbol{i}$, $\boldsymbol{i = 1, \ldots, S_b}$
for *G=1 to* $\boldsymbol{GMAX}$ **do**
  for *i=1 to* $\boldsymbol{S_b}$ **do**
    for *j=1 to* $\boldsymbol{N_c}$ **do**
      En el proceso quimiotáxico intercalar los nados propuestos con las Ecs. 7.1 y 7.2. Aplicar el operador de agrupamiento con la Ec. 3.4 usando $\boldsymbol{\beta}$ para la bacteria $\boldsymbol{\theta^i(j,G)}$
    end
  end
  if *G mod* $\boldsymbol{RepCycle}$ *== 0* **then**
    Realizar el proceso de reproducción ordenando la población de acuerdo a las reglas de factibilidad y eliminar a $\boldsymbol{S_r}$ peores bacterias y duplicar el resto de bacterias $\boldsymbol{S_b - S_r}$.
  end
  Realizar el proceso de eliminación-dispersión eliminando a la peor bacteria $\boldsymbol{\theta^w(j,G)}$ de la población actual considerando la técnica de manejo de restricciones.
  Actualizar el vector de tamaño de paso usando la Ec. 7.3
  if *G mod* $\boldsymbol{LS_G}$ *==0* **then**
    Aplicar SQP a la mejor bacteria de la población. Si la bacteria obtenida es mejor, ésta reemplaza a la mejor bacteria.
  end
end

Algoritmo 3: Pseudocodigo de TS-MBFOA.

7.2. TS-MBFOA en problemas de optimización númerica con restricciones

Los resultados de la propuesta Evolutiva MBFOA (TS-MBFOA) son presentados en esta sección, el algoritmo fue aplicado a los 24 problemas de prueba del CEC2006 (ver Anexo 8) y a tres problemas de diseño Mecatrónico. Un análisis de comportamiento de las modificaciones y mecanismo propuestos es presentado en el primer experimento, además, dos experimentos más son presentados para mostrar la superioridad de TS-MBFOA ante MBFOA, también se compara con los resultados obtenidos de diversos algoritmos de la literatura especializada.

7.2.1. TS-MBFOA en problemas de prueba

En los sucesivos experimentos los valores de los parámetros del algoritmo se encuentran en la Tabla 7.1. Para este primer experimento, el problema de prueba $\boldsymbol{g03}$ fue seleccionado debido a que su región factible es bastante pequeña y difícil de encontrar según la variable $\boldsymbol{\rho}$ de la Tabla 8.1 la cual indica que su valor es cerca de cero. Nuestro objetivo es observar el comportamiento de la propuesta al utilizar el mecanismo de sesgo, los operadores de nado, el tamaño de paso aleatorio, el incremento del número de bacterias en el proceso de eliminación-dispersión y el operador de búsqueda local. Los resultados de la mediana de las 30 ejecuciones independientes se utilizan para generar los gráficos de esta sección de análisis.

	Value	
Parámetros	**MBFOA**	**TS-MBFOA**
S_b	40	40
N_c	20	24
S_r	2	1
R	1.20E-03	5.00E-04
β	1.5	1.75
$RepCycle$	-	100
$GMAX$	value to reach Max_FEs	value to reach Max_FEs
LS_G	-	1 and ($GMAX$/2) generations
Max_LS		5,000 FEs
ss	-	8

Tabla 7.1: Parámetros del TS-MBFOA y MBFOA

En la figura 7.6 se muestra las bacterias generadas con el mecanismo de sesgo. A diferencia de MBFOA, TS-MBFOA obtuvo un grupo de bacterias con la suma de violación de restricción cerca de cero, sin embargo, una bacteria de MBFOA generada de forma aleatoria, obtuvo menor suma de violación de restricción. También se puede observar que las bacterias de MBFOA tienen un aumento progresivo o continuo en la suma de violación de restricción mientras que las bacterias de TS-MBFOA tienen un aumento brusco.

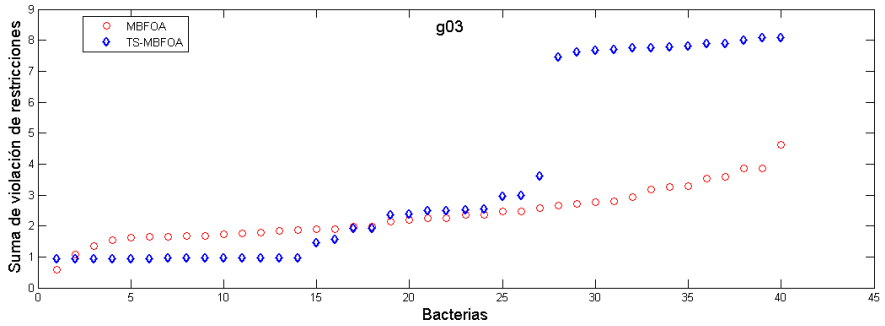

Figura 7.6: Población inicial de bacterias de MBFOA y TS-MBFOA para el problema de prueba ***g*03**.

Sin embargo, después del proceso quimiotáxico, a diferencia de MBFOA, TS-MBFOA es capaz de llevar las bacterias una la región factible. La figura 7.7 muestra este comportamiento, donde TS-MBFOA obtuvo algunas bacterias factibles y el resto de la población tiene una suma de violación de restricción cerca de cero. Es importante mencionar que la reproducción, los procesos de eliminación-dispersión y el operador SQP no se utilizaron en TS-MBFOA para este experimento. En el caso de MBFOA, una quimiotaxis, los procesos de reproducción y la eliminación-dispersión se llevaron a cabo sin generar soluciones factibles en esta primera generación del algoritmo.

Es claro que el proceso quimiotáxico, con los operadores de nado y el agrupamiento, ayudan a mejorar el rendimiento de TS-MBFOA, y esto se demuestra en la Figura 7.8 donde se calcula el número de nados exitosos por generación. Dos variantes de TS-MBFOA son presentados, una usando el operador de búsqueda local y otra sin este operador, y se comparan contra MBFOA. Ambas variantes tienen un comportamiento similar al obtener nados exitosos durante todas las generaciones, sin embargo el número de nados exitosos de MBFOA disminuyeron drásticamente en las primeras generaciones y nunca se incrementaron. Por otra parte, la tasa de nados exitoso (Sucessful swin rate) proporciona los siguientes resultados: TS-MBFOA con búsqueda local obtiene 5,05 %, TS-MBFOA sin búsqueda local obtiene 5,64 % y MBFOA obtiene solo el 0,46 %

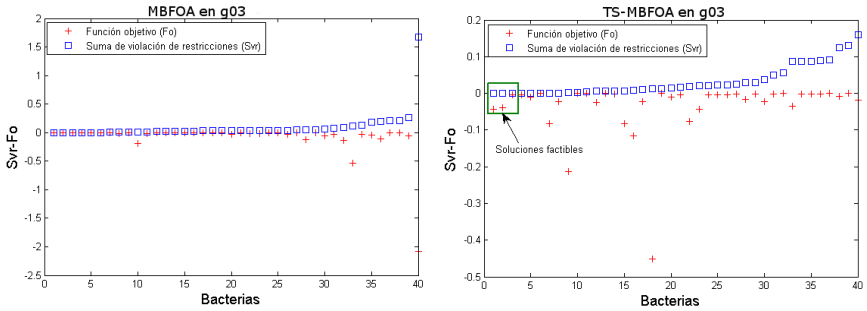

Figura 7.7: Población inicial de bacterias de MBFOA y TS-MBFOA después de una iteracción en el problema de prueba ***g*03**.

.

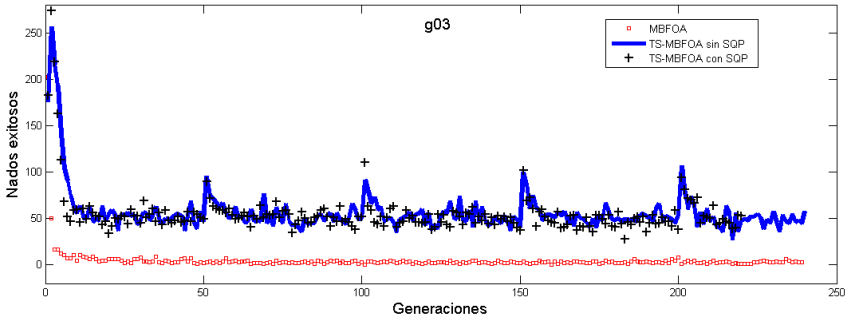

Figura 7.8: Nados exitosos obtenidos por MBFOA y dos versiones de TS-MBFOA en el problema de prueba ***g*03**.

La convergencia de la mediana de las 30 ejecuciones independientes de cada algoritmo se muestra en la Figura 7.9. La convergencia prematura es evidente en MBFOA porque cayó en un óptimo local en las primeras generaciones (ver gráfico á'). TS-MBFOA sin búsqueda local muestra una convergencia continua evitando caer en un óptimo local, es decir, evita la convergencia prematura. Sin embargo, esta propuesta necesitará más generaciones para encontrar el óptimo global (Ver gráfico 'b'). Por otro lado, TS-MBFOA con búsqueda local encontró el óptimo global para este problema de prueba desde la primera generación y lo mejoró en cada generación (véase el gráfico 'c'), este comportamiento se debe a la temprana intervención del operador de búsqueda local, es importante mencionar que la búsqueda local sólo se utiliza dos veces, una en la primera generación y otra en el ciclo medio de las generaciones. Además, la eliminación-dispersión de un grupo de bacterias y la reducción de la presencia del proceso de reproducción son importantes para evitar la convergencia prematura del algoritmo porque ambos permiten ampliar la exploración del espacio de búsqueda con la inserción de nuevas bacterias y al duplicar a la mejor bacteria en ciertas generaciones. En el gráfico de la letra 'd'se muestran las convergencias agrupadas de cada algoritmo.

En general, en este primer experimento se analizó el comportamiento del nuevo mecanismo y las modificaciones realizadas a MBFOA para mejorar su rendimiento en la propuesta TS-MBFOA, es claro que el mecanismo de sesgo permite generar bacterias cerca de la región factible y después del proceso quimiotáxico es probable encontrar soluciones factibles como fue en el caso del problema de prueba ***g*03**. Por otra parte, los nados exitosos se incrementan con los operadores de nado de

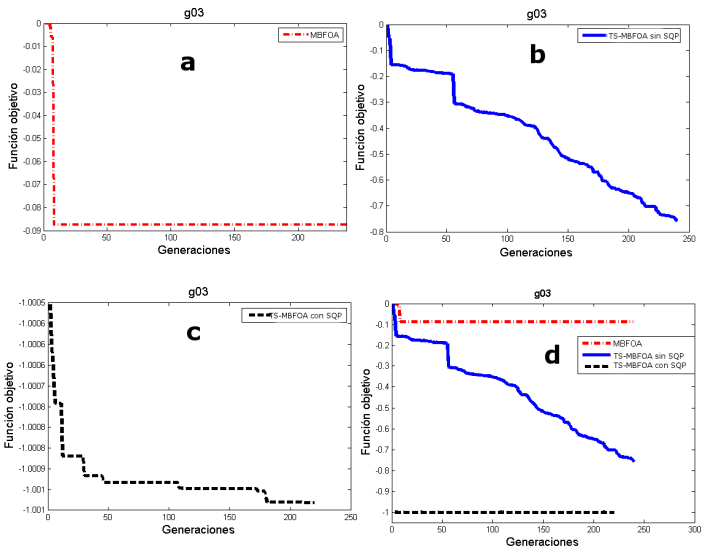

Figura 7.9: Convergencia de MBFOA y dos versiones de TS-MBFOA en el problema de prueba ***g*03**.

exploración y nado de explotación los cuales permiten una búsqueda rápida y eficaz. La convergencia prematura se reduce como se mostró en el gráfico de convergencias del problema de prueba ***g*03** donde las dos versiones de TS-MBFOA (con y sin buscador local) presentan un mejor comportamiento de convergencia que MBFOA.

Comparación de los resultados de TS-MBFOA contra MBFOA

En este segundo experimento se analizan los resultados obtenidos por TS-MBFOA los cuales son comparados contra MBFOA con el fin de observar el impacto del nuevo mecanismo y las modificaciones realizadas en TS-MBFOA para mejorar el rendimiento de MBFOA. En la Tabla 7.2 se presentan los resultados de ambos algoritmos y en negrita se remarcan los mejores resultados de ambos algoritmos, donde TS-MBFOA es el algoritmo que obtiene las mejores soluciones para el conjunto de funciones de prueba del CEC2006. La prueba de Wilcoxon con el **95 %** de confianza indica que hay una diferencia significativa a favor de TS-MBFOA en comparación contra MBFOA utilizando como criterio la mejor solución factible de cada problema de prueba.

En la Tabla 7.3 se presentan los resultados en términos de tasa de factibilidad, tasa de éxito y rendimiento exitoso. En cuanto a la tasa de factibilidad, ambos algoritmos encontraron soluciones factibles en la mayoría de los problemas de prueba, sin embargo, MBFOA no encontró soluciones factibles a 7 de los 24 problemas de prueba. Mientras tanto, TS-MBFOA no encontró soluciones factibles en los problemas de prueba ***g*20** y ***g*22**, sin embargo, en el resto de problemas de prueba el algoritmo encontró soluciones factibles en las 30 ejecuciones independientes. En términos de tasa de éxito de las soluciones factibles encontradas por ambos algoritmos en cada problemas de prueba, no todas satisfacen la condición de éxito, es decir, no todas las soluciones factibles son similares a la óptima global conocida; en esta medida, TS-MBFOA encontró el óptimo global en 20 de los 24 problemas de prueba y MBFOA soló 5 de los 24 problemas de prueba. Por último, la medida de rendimiento exitoso presenta el promedio de las evaluaciones que cada algoritmo necesita para encontrar una solución factible similar al óptimo global, en esta medida, TS-MBFOA necesita millones

Prob.	$f(\vec{x}^*)$	Criterios	MBFOA	TS-MBFOA	Prob.	$f(\vec{x}^*)$	MBFOA	TS-MBFOA
		Ejecuciones	25	25		25	25	
		WSRT	/	+		/	+	
g01	-15	Mejor	-9.88536	**-15**	g13	0.053941	-	**0.053942**
		Media	-4.91338	**-14.921**			-	0.3584
		Desv.Est.	2.11E+00	**2.97E-01**			-	2.81E-01
g02	-0.80362	Mejor	-0.44923	**-0.8036191**	g14	-47.7649	-42.5345	**-47.764888**
		Media	-0.33277	**-0.67934**			-38.6844	**-47.764888**
		Desv.Est.	4.70E-02	**6.23E-02**			2.52E+00	1.26E-13
g03	-1.0005	Mejor	-1.0004	**-1.001**	g15	961.715	961.7153	**961.71502**
		Media	**-1.0004**	-0.993			961.7177	**961.71502**
		Desv.Est.	**2.30E-05**	4.16E-02			1.57E-03	2.87E-05
g04	-30665.5	Mejor	-30665.1	**-30665.539**	g16	-1.90516	-1.90335	**-1.905155**
		Media	-30663.3	**-30665.539**			-1.88754	**-1.905155**
		Desv.Est.	4.19E+00	**1.17E-07**			5.64E-02	2.06E-15
g05	5126.497	Mejor	-	**5126.497**	g17	8853.54	-	8912.915
		Media	-	**5126.497**			-	8927.107
		Desv.Est.	-	**2.78E-08**			-	2.68E+00
g06	-6961.81	Mejor	-6960.83	**-6961.813875**	g18	-0.86603	-0.85966	**-0.866025**
		Media	-6950.8	**-6961.813875**			-0.73024	**-0.866025**
		Desv.Est.	1.21E+01	**4.63E-12**			1.18E-01	**5.08E-05**
g07	24.30621	Mejor	24.72605	**24.3062**	g19	32.65559	49.47301	**32.655592**
		Media	25.91026	**24.3062**			117.2929	**32.65581**
		Desv.Est.	8.36E-01	**1.20E-07**			7.33E+01	**6.40E-04**
g08	-0.09583	Mejor	**-0.095825**	**-0.095825**	g20	0.204979	-	-
		Media	**-0.095825**	**-0.095825**			-	-
		Desv.Est.	1.60E-12	**1.79E-17**			-	-
g09	680.6301	Mejor	680.6534	**680.63**	g21	193.7245	-	**193.72451**
		Media	680.706	**680.63**			-	**245.0109**
		Desv.Est.	4.25E-02	**5.24E-13**			-	**5.27E+01**
g10	7049.248	Mejor	7082.964	**7049.24802**	g22	236.431	-	-
		Media	7356.791	**7049.24802**			-	-
		Desv.Est.	4.89E+02	**1.27E-04**			-	-
g11	0.7499	Mejor	**0.7499**	**0.7499**	g23	-400.055	-	**-400.054**
		Media	**0.7499**	**0.7499**			-	**-377.089**
		Desv.Est.	2.12E-06	**1.86E-07**			-	**6.61E+01**
g12	-1	Mejor	-0.9999	**-1**	g24	-5.50801	**-5.508013**	**-5.508013**
		Media	-0.9992	**-1**			-5.50768	**-5.508013**
		Desv.Est.	1.95E-03	**0**			2.83E-04	**1.81E-15**

Tabla 7.2: Estadística básica de los resultados obtenidos por MBFOA y TS-MBFOA en el benchmark CEC2006.

de evaluación para encontrar el óptimo global en los problemas ***g*02** y ***g*21** debido a que estos problemas tienen restricciones activas, en el caso del problema ***g*21**, la relación estimada entre la región factible y el espacio de búsqueda es cercana a cero. Por otro lado, ***g*02** es un problema de prueba con 20 variables de decisión y su función es de tipo no lineal. TS-MBFOA tiene un pobre desempeño en los problemas de prueba ***g*17**, ***g*20**, ***g*22** y ***g*23**, estos problemas tienen como característica similar un gran número de restricciones activas.

De acuerdo con la medida de progreso obtenido por TS-MBFOA y MBFOA en los 24 problemas de prueba, TS-MBFOA es el algoritmo que mejora más la primera solución factible encontrada durante las generaciones (véase la Tabla 7.4) que MBFOA. En el caso de TS-MBFOA, un valor de medida de progreso no se encontró en los problemas de prueba ***g*20** y ***g*22**, y en el caso de MBFOA, un valor a la medida de progreso no se encontró en los problemas de prueba ***g*05**, ***g*13**, ***g*17**, $\boldsymbol{g20-23}$. También, en esta Tabla se puede observar que TS-MBFOA tiene un valor cercano de cero en los problemas de prueba ***g*05**, ***g*10**, ***g*11**, ***g*17**, ***g*21** y ***g*24** pero esto es debido a que las soluciones factibles encontradas por el algoritmo son similares a la óptima global desde primeras generaciones en la mayoría de estos problemas de prueba, excepto en ***g*17** y ***g*21** donde el algoritmo cayó en un óptimo local. Esta afirmación puede ser confirmada por los gráficos de convergencia de estos problemas de prueba (véase la Figura 7.10), la ejecución independiente encontrada en la mediana de las 30 realizadas fue utilizada para generar los gráficos.

Por otra parte, en los diagramas de caja (boxplot) de estos problemas de prueba (***g*05**, ***g*10**, ***g*11** y ***g*24**) la propuesta TS-MBFOA obtiene en las 30 ejecuciones independientes soluciones factibles

	Tasa de factibilidad		Tasa de éxito		Rendimiento exitoso	
Prob.	**MBFOA**	**TS-MBFOA**	**MBFOA**	**TS-MBFOA**	**MBFOA**	**TS-MBFOA**
g01	100 %	100 %	-	93 %	-	158,412
g02	100 %	100 %	-	4 %	-	3,239,750
g03	100 %	100 %	4 %	97 %	122,782	20,875
g04	100 %	100 %	-	100 %	-	8,206
g05	-	100 %	-	100 %	-	78,213
g06	100 %	100 %	-	100 %	-	15,145
g07	100 %	100 %	-	100 %	-	129,317
g08	100 %	100 %	12 %	100 %	68,024	1,024
g09	100 %	100 %	-	100 %	-	36,876
g10	100 %	100 %	-	87 %	210,201	102,552
g11	100 %	100 %	4 %	100 %	128,302	4,753
g12	100 %	100 %	12 %	100 %	-	6,953
g13	-	100 %	-	33 %	-	144,243
g14	33 %	100 %	-	100	-	81,638
g15	100 %	100 %	-	97 %	-	67,332
g16	100 %	100 %	-	100 %	-	43,415
g17	-	100 %	-	-	-	-
g18	90 %	100 %	-	93 %	-	177,578
g19	100 %	100 %	-	77 %	-	101,308
g20	-	-	-	-	-	-
g21	-	100	-	3	-	6,990,600
g22	-	-	-	-	-	-
g23	-	100 %	-	-	-	-
g24	100 %	100 %	32 %	100 %	20,400	1,782

Tabla 7.3: Tasa de factibilidad, tasa de éxito y rendimiento exitoso obtenido por MBFOA y TS-MBFOA en el benchmark CEC2006.

cercanas al óptimo global, excepto en los problemas de prueba ***g17*** y ***g21*** (ver Figuras 7.11, 7.14 y 7.15). En el caso de ***g17***, TS-MBFOA cae en un óptimo global desde las primeras generaciones. En ***g21*** TS-MBFOA comienza con soluciones no factibles, pero en las próximas generaciones es capaz de encontrar la región factible, sin embargo, el algoritmo no encuentra el óptimo global en la mayoría de las ejecuciones de este problema. El respectivo diagrama de caja de ***g21*** muestra que en varias ejecuciones TS-MBFOA obtiene soluciones factibles lejos de ser óptimo global.

	Mejor		Media		Desv.Est.	
Prob.	**MBFOA**	**TS-MBFOA**	**MBFOA**	**TS-MBFOA**	**MBFOA**	**TS-MBFOA**
g01	1.27E+00	1.48E+00	1.26E+00	1.40E+00	5.22E-03	5.49E-02
g02	1.60E+00	1.58E+00	1.60E+00	1.57E+00	1.88E-03	2.89E-03
g03	1.61E+00	3.22E+00	1.59E+00	1.68E+00	9.43E-03	4.06E-01
g04	5.17E+00	5.17E+00	5.17E+00	5.17E+00	6.02E-05	7.00E-16
g05	-	1.36E-07	-	1.36E-07	-	2.71E-12
g06	4.42E+00	4.71E+00	4.42E+00	4.45E+00	3.71E-03	6.61E-02
g07	0.00E+00	2.41E+00	0.00E+00	1.21E+00	0.00E+00	8.05E-01
g08	1.60E+00	1.60E+00	1.60E+00	1.60E+00	2.20E-05	2.07E-13
g09	0.00E+00	1.38E+00	0.00E+00	3.18E-01	0.00E+00	4.11E-01
g10	0.00E+00	5.04E-01	0.00E+00	1.30E-01	0.00E+00	1.59E-01
g11	6.38E-05	6.67E-05	2.51E-05	6.66E-05	2.83E-05	1.24E-07
g12	1.57E+00	1.57E+00	1.57E+00	1.57E+00	3.73E-10	4.21E-05
g13	-	1.46E+00	-	2.31E-01	-	5.01E-01
g14	1.86E+00	1.99E+00	1.84E+00	1.96E+00	2.04E-02	1.43E-02
g15	0.00E+00	7.79E-08	0.00E+00	6.87E-08	0.00E+00	1.49E-08
g16	1.96E+00	1.96E+00	1.84E+00	8.72E-01	1.42E-01	5.04E-01
g17	-	1.82E-02	-	1.46E-03	-	5.03E-03
g18	1.59E+00	1.68E+00	1.58E+00	1.59E+00	2.81E-03	2.55E-02
g19	0.00E+00	1.89E+00	0.00E+00	7.45E-01	0.00E+00	7.24E-01
g20	-	-	-	-	-	-
g21	-	7.99E-01	-	5.33E-01	-	1.74E-01
g22	-	-	-	-	-	-
g23	-	3.39E+00	-	3.06E+00	-	1.76E-01
g24	6.27E-01	6.85E-01	6.25E-01	6.29E-01	1.80E-03	1.05E-02

Tabla 7.4: Estadística básica de la medida de progreso obtenida por MBFOA y TS-MBFOA en el benchmark CEC2006.

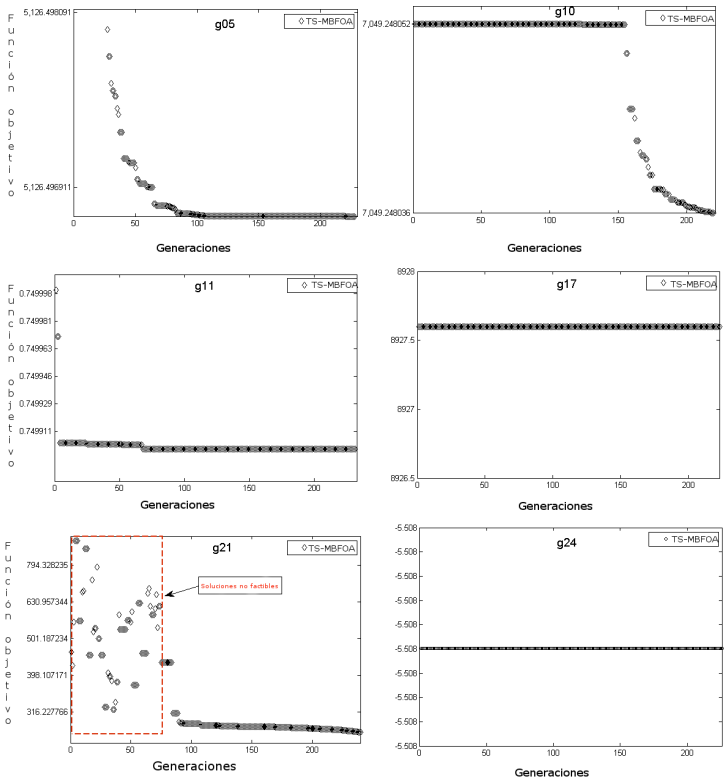

Figura 7.10: Convergencia de TS-MBFOA en los problemas de prueba ***g*05**, ***g*10**, ***g*11**, ***g*17**, ***g*21** y ***g*24**.

El diagrama de caja de cada problema de prueba se muestra en las Figuras 7.11, 7.12, 7.13, 7.14 y 7.15. En algunos problemas de prueba los algoritmos no encontraron soluciones factibles, por lo tanto, sus diagrama de caja no son presentados. Por otra parte, los diagramas de caja de los problemas de prueba se agruparon de acuerdo a la proximidad del óptimo global el cual es dibujado en cada caja con una línea punteada.

El diagramas de caja de TS-MBFOA muestran que en las 30 ejecuciones independientes este algoritmo encuentra la mismas soluciones factibles en la mayoría de los problemas de prueba, excepto en los problemas ***g*02**, ***g*13**, ***g*16**, ***g*21** y ***g*23**. En los problemas de prueba ***g*20** y ***g*22** TS-MBFOA y MBFOA no encontraron soluciones factibles, por lo tanto, sus diagramas de caja no fueron creados. Por otro lado, los resultados de MBFOA dibujan una caja grande en todos los problemas de prueba, es decir, en las 30 ejecuciones independientes el algoritmo encontró diferentes soluciones factibles, excepto en los problemas de prueba ***g*05**, ***g*13**, ***g*17**, ***g*20**, ***g*21**, ***g*22** y ***g*23** donde este algoritmo no encontró soluciones factibles.

En este experimento, TS-MBFOA mostró un alto rendimiento para resolver CNOPs y la capacidad no sólo para encontrar la región factible del espacio de búsqueda, incluso en una región muy pequeña, sino también para encontrar el óptimo global, como se mostró en el diagrama de caja del problema ***g*21** en la Figura 7.15. Por otra parte, es evidente la superioridad de TS-MBFOA sobre MBFOA para encontrar la región factible y mejorar las soluciones factibles encontradas.

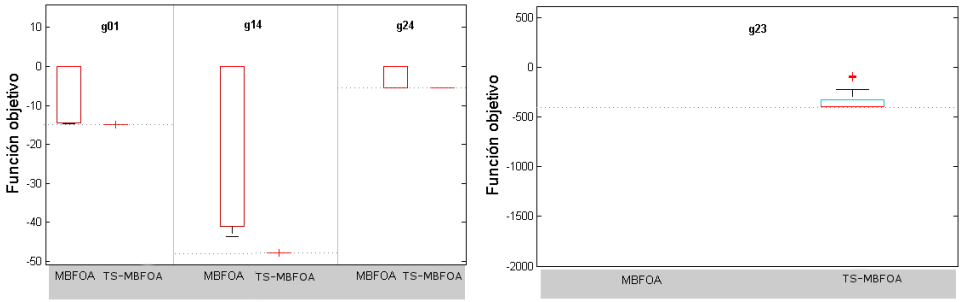

Figura 7.11: Diagrama de caja de la solución factible de cada ejecución independiente encontrada por TS-MBFOA en los problemas de prueba ***g*01**, ***g*14**, ***g*24** y ***g*23**.

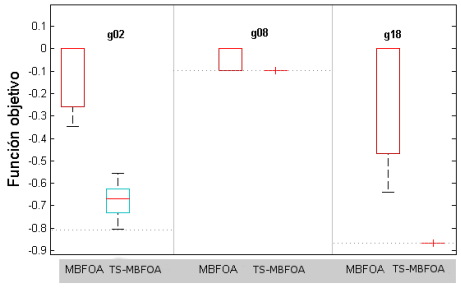

Figura 7.12: Diagrama de caja de la solución factible de cada ejecución independiente encontrada por TS-MBFOA en los problemas de prueba ***g*02**, ***g*08** y ***g*18**.

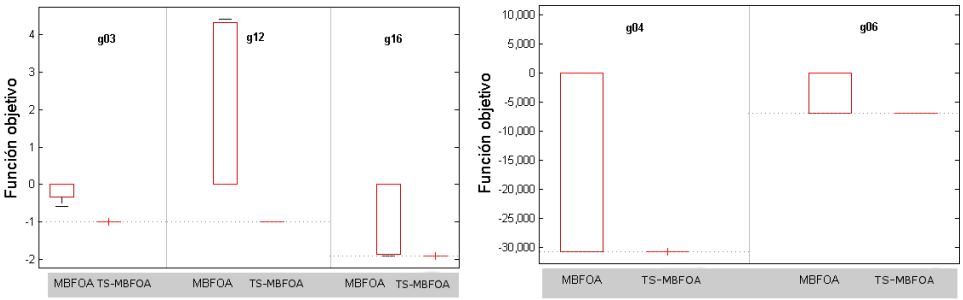

Figura 7.13: Diagrama de caja de la solución factible de cada ejecución independiente encontrada por TS-MBFOA en los problemas de prueba ***g*03**, ***g*12**, ***g*16**, ***g*04** y ***g*06**.

TS-MBFOA comparado contra algoritmos del estado del arte

En este tercer experimento el rendimiento de TS-MBFOA se compara con algunos algoritmos evolutivos como Mememtic Self-Adaptive Multi-Strategy Differential Evolution (Memetic-SAMSDE) [119], que es actualmente uno de los mejores algoritmo del estado del arte que resuelve el mismo conjunto de los problemas de prueba utilizados en esta sección. Este algoritmo también usa SQP como buscador local. A diferencia de TS-MBFOA que únicamente utiliza SPQ dos veces, en Memetic-SAMSDE se usa cada 50 generaciones. Además, Adaptive Penalty Formulation with GA (APF-GA) [120], Modified Differential Evolution (MDE) [118], Adaptive Tradeoff Model with evolution stra-

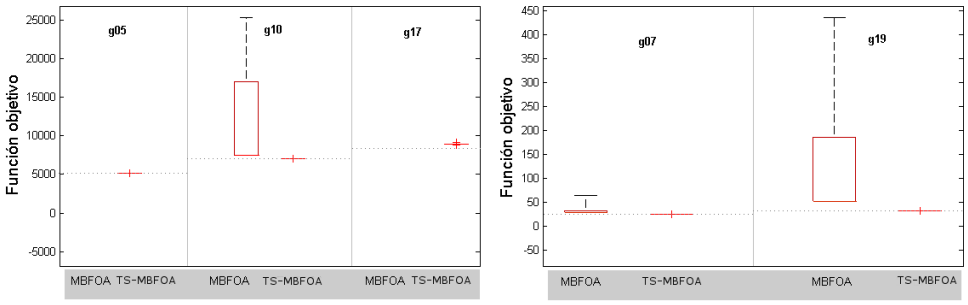

Figura 7.14: Diagrama de caja de la solución factible de cada ejecución independiente encontrada por TS-MBFOA en los problemas de prueba ***g*05**, ***g*10**, ***g*17**, ***g*07** y ***g*19**.

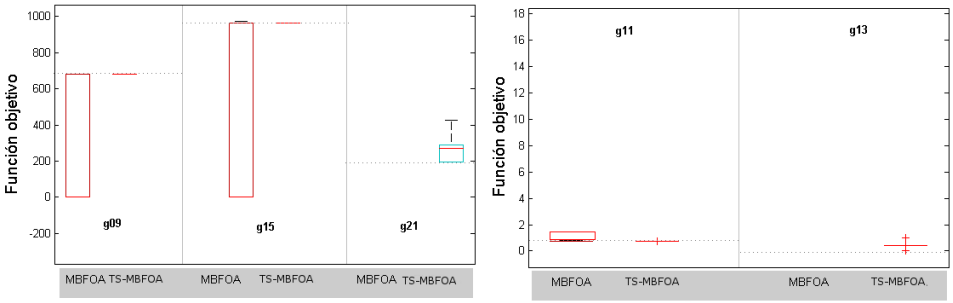

Figura 7.15: Diagrama de caja de la solución factible de cada ejecución independiente encontrada por TS-MBFOA en los problemas de prueba ***g*09**, ***g*15**, ***g*21**, ***g*11** y ***g*13**.

tegy (ATMES) [121], Multimembered evolution strategy (SMES) [122] y Stochastic ranking with evolution strategy (SR+ES) [98]. Incluso cuando ATMES, SMES y SR+ES resuelven únicamente los 13 primeros problemas de prueba, la comparación contra ellos es presentada.

El número de evaluaciones calculadas por APF-GA y MDE fue de 500,000. SR+ES utilizó 350,000 evaluaciones, mientras que para Memetic-SAMSDE, TS-MBFOA, ATMES y SMES se utilizaron 240,000 evaluaciones. Por otra parte, el número de ejecuciones independientes realizadas por TS-MBFOA, Memetic-SAMSDE, ATMES, PYMES y SR+ES fueron 30, y APF-GA y MDE realizaron 25 ejecuciones independientes.

Los resultados de los 24 problemas de prueba de todos los algoritmos son presentados en la Tabla 7.5 y la mejor solución de cada problema es remarcada en letras negritas.

Según los resultados obtenidos por cada algoritmo, la prueba de Wilcoxon con el 95 % de confianza sugieren que no hay diferencias significativas entre TS-MBFOA, Memetic-SAMSDE y MDE pero el último algoritmo utiliza 500,000 evaluaciones que son el doble de las evaluaciones utilizadas por TS-MBFOA. Por otro lado, hay una diferencia significativa a favor de TS-MBFOA cuando se compara contra APF-GA, ATMES, SMES y SR+ES, además, APF-GA y SR+ES utilizan un mayor número de evaluaciones que TS-MBFOA.

En este tercer experimento el rendimiento de TS-MBFOA se comparó contra seis algoritmos del estado del arte inspirados en la naturaleza utilizados para resolver CNOPs con éxito, con los resul-

Prob.	$f(\vec{x}^*)$	Criterios	TS-MBFOA	Memetic-SAMSDE	APF-GA	MDE	ATMES	SMES	SR+ES
g01	-15	Mejor	**-15**	**-15**	**-15**	**-15**	**-15**	**-15**	**-15**
		Media	-14.921	**-15**	**-15**	**-15**	**-15**	**-15**	**-15**
		Desv.Est.	2.97E-01	**0**	**0**	**0**	1.6E-14	**0**	**0**
g02	-0.8036191	Mejor	**-0.8036191**	**-0.8036191**	-0.803601	**-0.8036191**	-0.803339	-0.803601	-0.803515
		Media	-0.679341365	**-0.8036191**	-0.803518	-0.78616	-0.790148	-0.785238	-0.781975
		Desv.Est.	6.23E-02	**0**	1.00E-04	1.20E-02	1.30E-02	1.67E-02	2.00E-02
g03	-1.0005001	Mejor	**-1.001**	-1.0005	**-1.001**	-1.0005	-1	-1	-1
		Media	-0.993	-1.0005	**-1.001**	-1.0005	-1	-1	-1
		Desv.Est.	4.16E-02	**0**	**0**	**0**	5.90E-05	0.0000209	0.0000209
g04	-30665.5387	Mejor	**-30665.539**	**-30665.539**	**-30665.539**	**-30665.5386**	**-30665.539**	**-30665.539**	**-30665.539**
		Media	**-30665.539**	**-30665.539**	**-30665.539**	**-30665.5386**	**-30665.539**	**-30665.539**	**-30665.539**
		Desv.Est.	1.17E-11	**0**	1.00E-04	1.00E-04	7.4E-12	**0**	2.00E-05
g05	5126.49671	Mejor	**5126.497**	**5126.497**	**5126.497**	**5126.497**	5126.498	5126.599	**5126.497**
		Media	**5126.497**	**5126.497**	5127.5423	**5126.497**	5127.648	5174.492	5128.881
		Desv.Est.	2.78E-08	**0**	1.43E+00	**0**	1.80E+00	5.01E+01	3.50E+00
g06	-6961.81388	Mejor	**-6961.813875**	**-6961.813875**	**-6961.814**	**-6961.814**	**-6961.814**	**-6961.814**	**-6961.814**
		Media	**-6961.813875**	**-6961.813875**	**-6961.814**	**-6961.814**	**-6961.814**	-6961.284	-6875.94
		Desv.Est.	4.63E-12	**0**	**0**	**0**	4.6E-12	1.85E+00	1.60E+02
g07	24.3062091	Mejor	**24.3062**	**24.3062**	**24.3062**	**24.3062**	**24.306**	24.327	24.307
		Media	**24.3062**	**24.3062**	**24.3062**	**24.3062**	24.316	24.475	24.374
		Desv.Est.	1.20E-07	**0**	**0**	**0**	1.10E-02	1.32E-01	6.60E-02
g08	-0.095825	Mejor	**-0.095825**	**-0.095825**	**-0.095825**	**-0.095825**	**-0.095825**	**-0.095825**	**-0.095825**
		Media	**-0.095825**	**-0.095825**	**-0.095825**	**-0.095825**	**-0.095825**	**-0.095825**	**-0.095825**
		Desv.Est.	1.79E-17	**0**	**0**	**0**	2.8E-16	**0**	**0**
g09	680.630057	Mejor	**680.63**	**680.63**	**680.63**	**680.63**	**680.63**	680.632	**680.63**
		Media	**680.63**	**680.63**	**680.63**	**680.63**	680.639	680.643	680.656
		Desv.Est.	5.24E-13	**0**	**0**	**0**	1.00E-02	1.55E-02	3.40E-02
g10	7049.24802	Mejor	**7049.24802**	**7049.24802**	**7049.24802**	**7049.24802**	7052.253	7051.903	7051.903
		Media	**7049.24802**	**7049.24802**	7077.6821	**7049.24802**	7250.437	7253.047	7253.047
		Desv.Est.	1.27E-04	**0**	5.12E+01	**0**	1.20E+02	1.36E+02	1.36E+02
g11	0.7499	Mejor	**0.7499**	**0.7499**	**0.7499**	**0.7499**	**0.75**	**0.75**	**0.75**
		Media	**0.7499**	**0.7499**	**0.7499**	**0.7499**	**0.75**	**0.75**	**0.75**
		Desv.Est.	1.86E-07	**0**	**0**	**0**	3.40E-04	1.52E-04	8.00E-05
g12	-1	Mejor	**-1**	**-1**	**-1**	**-1**	**-1**	**-1**	**-1**
		Media	**-1**	**-1**	**-1**	**-1**	**-1**	**-1**	**-1**
		Desv.Est.	**0**	**0**	**0**	**0**	1.00E-03	**0**	**0**
g13	0.053941	Mejor	**0.053942**	**0.053942**	**0.053942**	**0.053942**	0.05395	0.053986	0.067543
		Media	0.358400084	**0.053942**	**0.053942**	**0.053942**	0.053959	0.166385	0.067543
		Desv.Est.	2.81E-01	**0**	**0**	**0**	1.30E-05	1.77E-01	3.10E-02
g14	-47.764888	Mejor	**-47.764888**	**-47.764888**	-47.76479	-47.764887	-	-	-
		Media	**-47.764888**	**-47.764888**	-47.76479	-47.764874	-	-	-
		Desv.Est.	1.26E-13	**0**	1.00E-04	1.40E-05	-	-	-
g15	961.715022	Mejor	**961.71502**	**961.71502**	**961.71502**	**961.71502**	-	-	-
		Media	**961.71502**	**961.71502**	**961.71502**	**961.71502**	-	-	-
		Desv.Est.	2.87E-05	**0**	**0**	**0**	-	-	-
g16	-1.905155	Mejor	**-1.905155**	**-1.905155**	**-1.905155**	**-1.905155**	-	-	-
		Media	**-1.905155**	**-1.905155**	**-1.905155**	**-1.905155**	-	-	-
		Desv.Est.	2.06E-15	**0**	**0**	**0**	-	-	-
g17	8853.53967	Mejor	8912.914588	**8853.5397**	8853.5398	**8853.5397**	-	-	-
		Media	8927.107321	**8823.5397**	8888.4876	**8853.5397**	-	-	-
		Desv.Est.	2.68E+00	**0**	2.90E+01	**0**	-	-	-
g18	-0.866025	Mejor	**-0.866025**	**-0.866025**	**-0.866025**	**-0.866025**	-	-	-
		Media	**-0.866025**	**-0.866025**	**-0.865925**	**-0.866025**	-	-	-
		Desv.Est.	5.08E-05	**0**	1.00E-04	**0**	-	-	-
g19	32.655592	Mejor	**32.655592**	**32.655593**	**32.655593**	**32.64827**	-	-	-
		Media	32.65581447	**32.655593**	**32.655593**	33.34125	-	-	-
		Desv.Est.	6.40E-04	**0**	**0**	8.47E-01	-	-	-
g20	0.2049794	Mejor	-	-	-	-	-	-	-
		Media	-	-	-	-	-	-	-
		Desv.Est.	-	-	-	-	-	-	-
g21	193.72451	Mejor	**193.72451**	**193.72451**	196.63301	**193.72451**	-	-	-
		Media	245.010861	**193.72451**	199.51581	**193.72451**	-	-	-
		Desv.Est.	5.27E+01	**0**	2.36E+00	**0**	-	-	-
g22	236.430976	Mejor	-	**236.370313**	-	-	-	-	-
		Media	-	**245.738829**	-	-	-	-	-
		Desv.Est.	-	**9.05E+00**	-	-	-	-	-
g23	-400.0551	Mejor	-400.0544473	**-400.0551**	-399.7624	**-400.0551**	-	-	-
		Media	-377.0888655	**-400.0551**	-394.7627	**-400.0551**	-	-	-
		Desv.Est.	6.61E+01	**0**	3.87E+00	**0**	-	-	-
g24	-5.50801327	Mejor	**-5.508013**	**-5.508013**	**-5.508013**	**-5.508013**	-	-	-
		Media	**-5.508013**	**-5.508013**	**-5.508013**	**-5.508013**	-	-	-
		Desv.Est.	1.81E-15	**0**	**0**	**0**	-	-	-

Tabla 7.5: Estadística básica de los resultados obtenidos por TS-MBFOA y otros algoritmos en el benchmark CEC2006.

tados de la comparación se prueba que TS-MBFOA es un algoritmo competitivo.

7.2.2. TS-MBFOA en problemas de diseño Mecatrónico

La propuesta TS-MBFOA fue probada en tres problemas de diseño Mecatrónico derivados de la Síntesis de un mecanismo plano de cuatro barras los cuales se utilizan ampliamente en el diseño de maquinaria, ya que son los mecanismos articulados más simple para el movimiento controlado con un grado de libertad. La síntesis de estos mecanismos es un CNOP bien conocido, y se utilizan

originalmente dos enfoques clásicos para esta síntesis: métodos gráficos y analíticos. Sin embargo, la implementación de este tipo de soluciones es un tema complicado y sus resultados son bastante limitados; por esta razón el diseño de estos mecanismos es un caso de optimización numérico duro. Hay varios tipos de síntesis; en este trabajo se aborda el diseño dimensional de un mecanismo, es decir, calcular la longitud de los enlaces necesarios para la generación de un movimiento específico [123]. Aunque en la literatura existen metaheurísticas como el Algoritmo Genético [124], Evolución Diferencial [125], Recocido Simulado [126], PSO [127], Colonia Artificial de Abjeas [128] y Búsqueda de armonía [129] que realizan la síntesis de mecanismos de cuatro barras BFOA o propuestas derivadas de este algoritmo, estos no han sido utilizados para resolver estos problemas.

Los tres casos de la síntesis del mecanismo plano de cuatro barras a resolver son: 1) El segumiento de la trayectoria lineal del mecanismo de cuatro barras, 2) el seguimiento de la trayectoria no lineal del mismo mecanismo y 3) la generación de movimiento delimitado por un conjunto con pares de puntos, es decir, posición de control. Una descripción a detalle de cada problema es encontrado en el Capítulo 8 en la Sección 8, en la Tabla 7.6 se encuentra un breve resumen de las características de cada problema donde Max_FEs es el número máximo de evaluaciones permitidas y n es el número de variables. Además en la Sección 8 una descripción detallada de los casos es presentada.

Problem	Max_FEs	n	**Objective type**
M01	500,000	15	No separable
M02	100,000	6	No separable
M03	500,000	19	No separable

Tabla 7.6: Resumen de los 3 problemas del sistema de cuatro barras.

Parámetros	**TS-MBFOA**	**MBFOA**
S_b	60	40
N_c	10	24
S_r	1	1
ss	8	-
β	1.75	1.75
RepCycle	100	100
GMAX	value to reach Max_Evals	value to reach Max_FEs

Tabla 7.7: Parámetros de TS-MBFOA y MBFOA.

El valor de los parámetros utilizados para resolver cada uno de los tres problemas es presentado en la Tabla 7.7, el valor de los parámetros de TS-MBFOA fueron obtenidos con la herramienta iRace [130]. El valor de los parámetros de MBFOA fueron tomados de [131].

Es importante mencionar que TS-MBFOA para el conjunto de problemas de diseño Mecatrónico no usará SQP debido a que ciertos valores de las funciones o restricciones de los tres casos se pueden indefinir por la información de segundo orden que ocupa el método SQP en sus iteraciones, por lo tanto el parámetro $\boldsymbol{LS_G}$ es eliminado. El parámetro fijo fue $\boldsymbol{GMAX}$ relacionado con la condición de termino de $\boldsymbol{Max\_FEs}$ considerado en cada problema del sistema de cuatro barras.

Comparación entre MBFOA y TS-MBFOA

Los resultados obtenido por MBFOA y TS-MBFOA son presentados y comparados en la Tabla 7.8 en términos de mejor valor, valores de media y desviación estándar obtenidos de las 30 ejecuciones

independientes. Además, la Tabla 7.9 presenta los resultados de ambos algoritmos en términos de tasa de factibilidad.

Basado en estos resultados, ambos algoritmos encuentran soluciones factibles a todos los problemas de optimización de la síntesis de cuatro barras (ver Tabla 7.9). Sin embargo, los resultados obtenidos por TS-MBFOA fueron mejores que MBFOA de acuerdo al valor de las estadísticas resportadas. La prueba de WSRT con el 95 % de confidencialidad obtiene una diferencia significativa entre ambos algoritmos.

Prob.	Criterios	MBFOA	TS-MBFOA
	Evaluaciones	500,000	500,000
M01	Mejor	1.20E+00	**1.26E-29**
	Media	25.00E+00	**2.40E-02**
	Desv. Est.	2.57E+01	**9.15E-02**
	Evaluaciones	100,000	50,000
M02	Mejor	0.002997125	**0.002628079**
	Media	3.69E+00	**2.63E-03**
	Desv. Est.	2.98E-04	**1.31E-17**
	Evaluaciones	500,000	500,000
M03	Mejor	0.5630512	**0.2750193**
	Media	16.56E+00	**1.07E+00**
	Desv. Est.	2.72E+01	**1.10E+00**

Tabla 7.8: Estadística básica de los resultados obtenidos por MBFOA y TS-MBFOA de los tres problemas de optimización de la síntesis del mecanismo plano de cuatro barras.

Prob.	MBFOA	TS-MBFOA
M01	100 %	100 %
M02	100 %	100 %
M03	100 %	100 %

Tabla 7.9: Tasa de factibilidad obtenida por MBFOA y TS-MBFOA de los tres problemas de optimización de la síntesis del mecanismo plano de cuatro barras.

La gráfica de convergencia de cada problema de optimización son mostradas en las Figuras 7.16, 7.17 y 7.18 usando la ejecución localizada en el valor de la mediana de las 30 ejecuciones independientes.

En el problema ***M*01**, MBFOA converge en una solución óptima local. Por otra parte, TS-MBFOA es hábil para obtener una mejor solución factible. En el problema ***M*02**, MBFOA presenta una convergencia más rápida que TS-MBFOA. Sin embargo, MBFOA queda atrapado en un óptimo local mientras que TS-MBFOA tiene la habilidad de mejorar la solución. Finalmente, en el problema ***M*03** ambos algoritmos inician con un similar comportamiento, pero TS-MBFOA tiene la habilidad de encontrar una mejor solución en la mitad final de la búsqueda.

Los detalles de la solución localizada en el valor de la mediana de las 30 ejecuciones independientes son presentados en la Tabla 7.10. Basados en las diferencias observadas en los valores de las variables de decisión, es claro que TS-MBFOA es hábil para explorar otras áreas del espacio de búsqueda con mejores valores factibles en la funcion objetivo.

Con este primer experimento se puede concluir que TS-MBFOA fue hábil para obtener mejores resultados que MBFOA evitando óptimos locales en los tres casos de la sístesis del mecanismo plano de cuatro barras.

	M01		M02		M03	
Variables	MBFOA	TS-MBFOA	MBFOA	TS-MBFOA	MBFOA	TS-MBFOA
X_1	57.92342335	32.38804184	36.94081373	14.3145474	45.12424927	59.96966446
X_2	14.25129442	7.691176933	2.03539868	2.21116581890101	4.354984633	0.436956203
X_3	41.60988997	26.61191672	35.5742863	14.3145473987279	35.40040087	7.684578291
X_4	47.9591382	30.3913811	36.6889199298196	14.3145473987279	40.73333805	54.67738769
X_5	-51.0832316	37.84886272	2.33089990346102	2.17436158983768	37.1230094	-1.005302961
X_6	-23.78710654	19.69599228	-0.147913852355051	0.0222097752927143	54.07107216	-8.325939024
X_7	2.657985522	4.081991482			0.051061904	1.31E-05
X_8	-36.18342533	-12.34670526			35.18352501	-4.003143707
X_9	1.590148428	58.33436734			-49.2073547	8.911787419
X_{10}	2.662155089	1.541782762			0.051708269	1.32E-07
X_{11}	2.937908128	2.38381157			4.31038762	0.075478987
X_{12}	3.207941546	2.919441622			4.340261456	0.686220008
X_{13}	3.425916221	3.428688234			4.345117555	1.385316707
X_{14}	3.592923694	3.980362235			4.413690541	1.979526863
X_{15}	3.748942027	5.012387829			4.441375824	2.558519198
X_{16}					5.127114977	3.331787694
X_{17}					5.482639033	3.725623295
X_{18}					5.505238975	4.083026763
X_{19}					5.58273151	4.833519954
$f(\vec{x})$	15.26227802	**1.23E-18**	0.00371541960656681	**0.002628079**	66.15179375	**0.479623975**

Tabla 7.10: Detalles de la solución encontrada por MBFOA y TS-MBFOA en la ejecución localizada en el valor de la mediana de las 30 ejecuciones independientes.

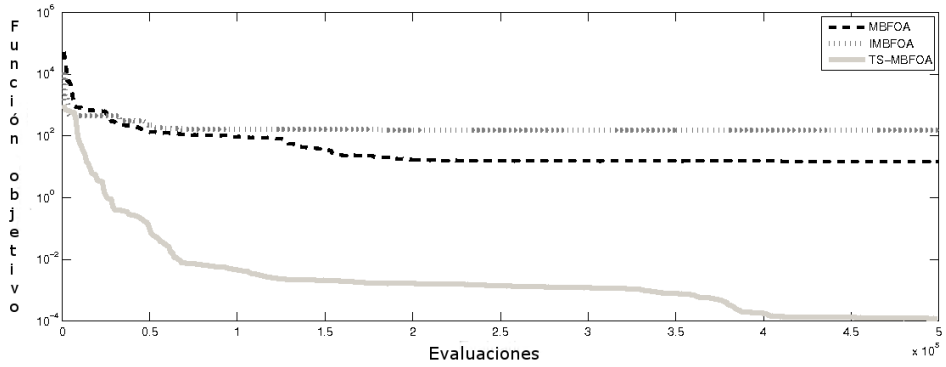

Figura 7.16: Gráfica de convergencia de MBFOA y TS-MBFOA en el problema $\boldsymbol{M01}$.

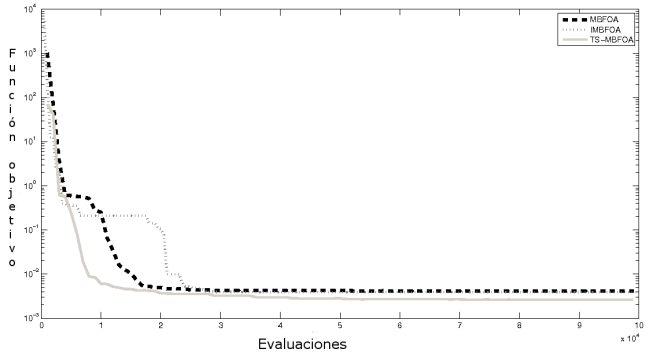

Figura 7.17: Gráfica de convergencia de MBFOA y TS-MBFOA en el problema $\boldsymbol{M02}$.

Comparación entre TS-MBFOA y un Algoritmo Evolutivo para diseño mecánico

Los resultados de TS-MBFOA son comparados contra una propuesta basada en Evolución Diferencial (ED) diseñada para resolver problemas de diseño mecánico [132]. Los parámetros usados por ED fueron: 100 individuos y 5000 generaciones en los problemas $\boldsymbol{M01}$ y $\boldsymbol{M03}$. Para el problema $\boldsymbol{M02}$ este algoritmo uso 100 individuos y 1000 generaciones. Los valores de $\boldsymbol{F}$ y $\boldsymbol{CR}$ fueron generados aleatoriamente en cada generación entre los intervalos: [0.3, 0.9] y [0.8, 1.0], respectivamente.

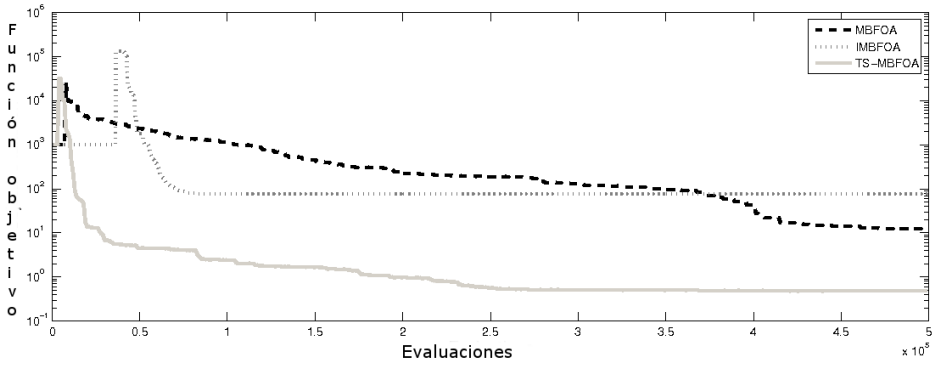

Figura 7.18: Gráfica de convergencia de MBFOA y TS-MBFOA en el problema ***M*01**.

La Tabla 7.11 incluye los resultados estadísticos de las 30 ejecuciones independientes realizadas por TS-MBFOA y ED a los tres problemas de optimización de la síntesis de cuatro barras. En las 30 ejecuciones ambos algoritmos encontraron soluciones factibles, además, la prueba WSRT con el 95 % de confidencialidad indica que la diferencia en los resultados finales no fueron significativos entre los algoritmos. Sin embargo, TS-MBFOA requiere menos evaluaciones para encontrar tales resultados en el problema ***M*01** (500,000 contra 750,000 en el problema ***M*01**).

Prob.	Criterios	TS-MBFOA	ED
	Evaluaciones	500,000	750,000
M01	Mejor	**1.26217E-29**	**1.26218E-29**
	Media	2.40E-02	**1.99E-03**
	Des. Est.	9.15E-02	**5.39E0-03**
	Evaluaciones	100,000	100,000
M02	Mejor	**0.002628079**	**0.002628079**
	Media	**2.63E-03**	2.63E-03
	Desv. Est.	**1.31E-17**	4.473E-10
	Evaluaciones	500,000	500,000
M03	Mejor	0.2750193	**0.274968745**
	Media	**1.07E+00**	1.31E+00
	Desv. Est.	**1.10E+00**	3.27E+00

Tabla 7.11: Estadística básica de los resultados obtenidos por TS-MBFOA y ED de los tres problemas de optimización de la síntesis del mecanismo plano de cuatro barras.

En este segundo experimento se observa que TS-MBFOA obtiene resultados similares a los encontrados por el algoritmo basado en ED para resolver problemas de optimización de diseño mecánico. Sin embargo, TS-MBFOA requiere menos evaluaciones para encontrar tales resultados en dos de los tres problemas de optimización de la sístesis de cuatro barras.

Simulación de las mejores soluciones obtenidas por MBFOA, TS-MBFOA y ED

Este experimento final se compara la mejor solución encontrada por MBFOA, TS-MBFOA y la versión de ED en los tres problemas usando simulación y análisis desde le punto de vista mecánico. Los valores correspondientes para las variables de decisión de cada problema son presentados en la Tabla 7.12.

Como se muestra en la Figuras 7.19 y 7.21 para los problemas ***M*01** y ***M*03**, respectivamente, los mecanismos sintetizados con TS-MBFOA y ED tiene un rendimiento similar, mientras que el

mecanismo optimizado por MBFOA es menos eficiente. Aunque los tres mecanismos pasan sobre la trayectoria especificada por los seis puntos de presición, en ***M*01** el bucle de recuperación para iniciar de nuevo el seguimiento es mucho más largo con el diseño de MBFOA; en términos de consumo de tiempo y energía, este seguimiento más largo representa una desventaja en relación con los otros mecanismos diseñados por TS-MBFOA y ED.

Para el problema ***M*02**, los resultados de los tres algoritmos son comparables desde el punto de vista mecánico, puesto que las dimensiones de sus elementos mecánicos varían en menos del 25 % entre ellos, produciendo mecanismos similares.

Para el problema de M03, los tres mecanismos cubren de una manera similar la ruta especificada por los pares de puntos de precisión. Sin embargo, los datos de la Tabla 7.12 indican que las barras 1 y 4 son considerablemente más larga en el mejor mecanismo producido por MBFOA que en los mejores diseños de los otros dos algoritmos. Esto podría producir una menor eficiencia en el rendimiento del sistema originada por los efectos no deseados en estos enlaces. Tales efectos pueden ser: una mala alineación y un equilibrio débil debido a una flexión transversal producido por las cargas internas y una tensión más grande entre los elementos mecánicos en los puntos de apriete, relacionado con el ángulo de transmisión del mecanismo que, en general, es una medida de su calidad.

	M01			**M02**			**M03**		
Variables	**MBFOA**	**TS-MBFOA**	**ED**	**MBFOA**	**TS-MBFOA**	**ED**	**MBFOA**	**TS-MBFOA**	**ED**
x_1	56.96046969	24.0706441	37.35322401	19.59633374	14.31454786	14.31436975	38.12083651	2.632965423	2.614591501
x_2	4.060834527	7.250066369	8.414037276	2.140600536	2.211165812	2.211165657	0.774854018	1.056973251	1.034801583
x_3	8.923787074	20.94483824	27.79863521	18.56633857	14.31454786	14.3143696	19.29477637	1.763888952	1.826884381
x_4	56.08270471	22.57639254	37.00803944	18.5842461	14.31454786	14.31436974	24.68930882	2.205482703	2.207891726
x_5	-25.83505865	31.53789197	37.61267067	2.239379687	2.174361597	2.174363923	8.333794141	1.206113171	1.250924301
x_6	-29.75007805	16.20250155	16.97308561	0.023572299	0.022209768	0.022212213	-6.463530675	0.363242643	0.447340178
x_7	4.267457973	4.156218625	3.966832786				5.93886	5.813750828	5.826803696
x_8	54.8103088	-7.286682385	-9.678596639				-8.625076191	0.084076204	0.099169603
x_9	36.62395408	52.85173965	59.18270796				6.384329872	1.437990445	1.328798537
x_10	1.120012858	1.24	1.717902384				0.038575927	0.396045164	0.410281221
x_11	1.362491164	2.309698447	2.451658775				0.42340301	1.029004626	1.039364984
x_12	1.621239336	2.879833244	2.966271497				0.971029032	1.625114804	1.650008311
x_13	1.855988457	3.422015391	3.464656513				1.704933689	2.219097833	2.260034392
x_14	2.11850087	3.997488978	4.013579749				2.378218281	2.803908752	2.865981127
x_15	2.453816162	5.243554444	5.124608233				3.063472839	3.403882079	3.490250618
x_16							3.67844246	4.09753211	4.163941276
x_17							3.938125257	4.853550562	4.905546281
x_18							4.286898695	5.380182303	5.416480223
x_19							5.042731775	6.052924519	6.06760838
$f(\vec{x})$	1.206824913	**1.26217E-29**	**1.26218E-29**	0.002997125	**0.002628079**	**0.002628079**	0.5630512	0.275019399	**0.274968745**

Tabla 7.12: Mejor solución encontrada por MBFOA, TS-MBFOA y ED para cada uno de los problemas de la síntesis de cuatro barras.

A partir de este último experimento, las simulaciones de las mejores soluciones sugieren que TS-MBFOA fue capaz de encontrar, desde un punto de vista mecánico, mecanismos de cuatro barras de alta calidad en los tres casos presentados en este trabajo. Particularmente en el problema ***M*01**, TS-MBFOA fue capaz de encontrar una solución muy competitiva con menos evaluaciones con respecto a un algoritmo de evolución diferencial diseñado para resolver problemas de diseño mecánico.
En general:

- TS-MBFOA es capaz de encontrar soluciones factibles en una región grande o pequeña del espacio de búsqueda desde las primeras generaciones del algoritmo, es decir, utilizando un mínimo de evaluaciones de costo

- Los experimentos demuestran que TS-MBFOA es mejor que MBFOA y competitivo contra los algoritmos del estado del arte tanto en problemas de prueba como en problemas reales, en algunos casos la propuesta requiere de menos evaluaciones

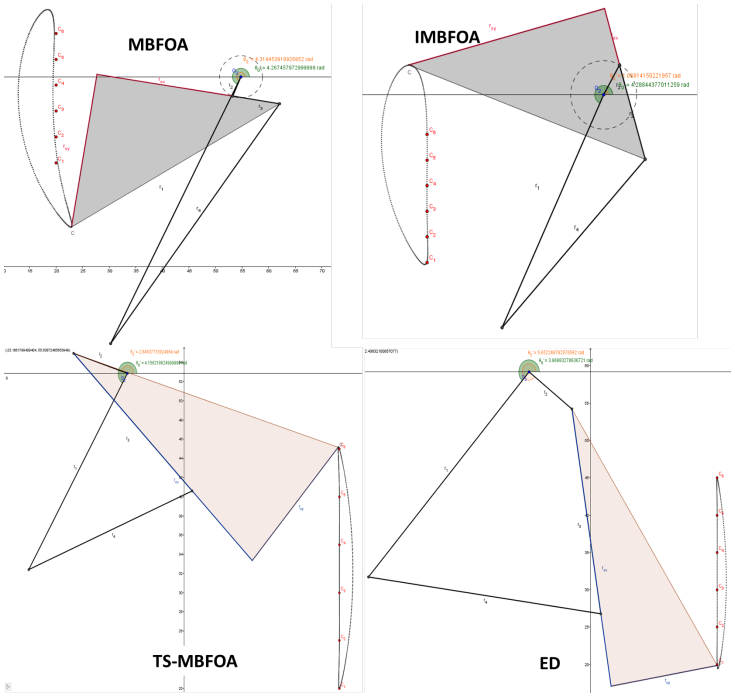

Figura 7.19: Simulación de la mejor solución para el problema ***M*01**.

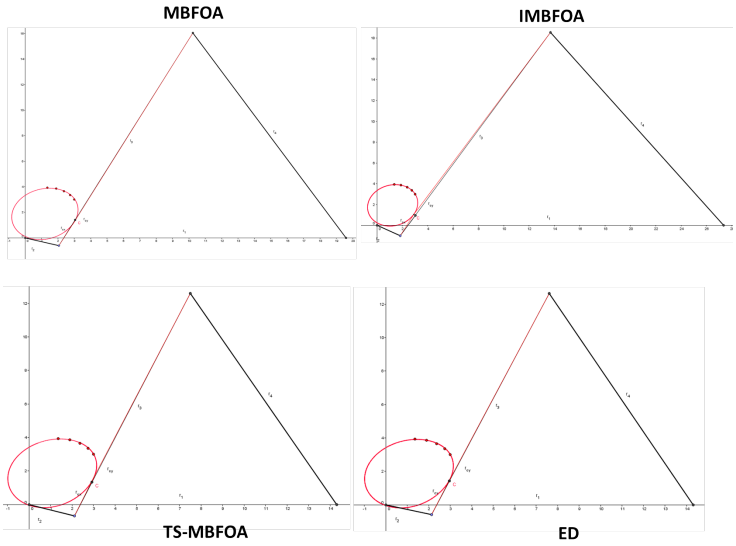

Figura 7.20: Simulación de la mejor solución para el problema ***M*02**.

- Los operadores de nado propuestos junto con el tamaño de paso aleatorio permiten una mejor exploración y explotación del espacio de búsqueda
- El mecanismo de sesgo permiten que el algoritmo desde sus primeras generaciones obtenga soluciones factibles
- La diversidad de la población fue conservada en lapsos de tiempo con el incremento del número de bacterias a eliminar-dispersar
- Por último, para el conocimiento de los autores, este es el primer intento de diseñar un algoritmo

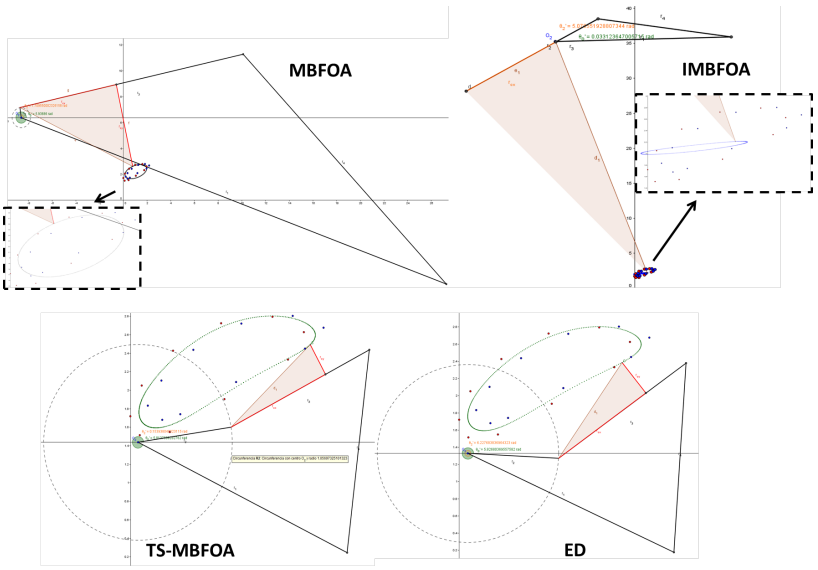

Figura 7.21: Simulación de la mejor solución para el problema $\boldsymbol{M03}$.

basado en BFOA que utiliza mutación como operador de natación

- Además, en los tres problemas de optimización de la síntesis de cuatro barras TS-MBFOA obtuvo resultados altamente competitivos y mejores que los obtenidos por MBFOA en los problemas $\boldsymbol{M01}$ y $\boldsymbol{M03}$.

7.2.3. TS-MBFOA generando menús nutricionales

La propuesta TS-MBFOA fue utilizada para generar menús nutricionales con algunas adaptaciones menores. El modelado del problema para la generación de menús nutricionales es detallado en el Anexo 8 en la Sección refsec:pmr.

Existen generadores de menús nutricionales que utilizan sistemas expertos como Diet creator [133] y Nutrimind [134] los cuales son de uso comercial. Además, el algoritmo genético se ha implementado para generar menús nutricionales Con base en una modelación matemática, un ejemplo de ello es la propuesta en [135] donde se busca maximizar la variedad de submenús en una dieta. Sin embargo, no existe reporte de alguna propuesta basada en BFOA que se haya utilizado para generar menús nutricionales, por lo tanto, esta será la primera vez que se utilice forrajeo de bacterias para generar dichos menús.

Adaptación de TS-MBFOA para generar menús nutricionales

TS-MBFOA fue adaptado para generar la población inicial con variables enteras dentro del rango del número de alimentos existentes en cada grupo de la base de alimentos y en el proceso quimiotáctico la solucion de cada bacteria es redondeada al entero más proximo debido a que los valores del vector $\vec{\boldsymbol{k}}$ hacen referencia a indexs de la base de alimentos. Una solución para este problema es representado de la siguiente manera:

$\boldsymbol{\theta^i(j,G)}$=$\{18, 48, 25, 19, 2, 47\}, \{17, 43, 23\}, \{4, 39, 19, 19, 28, 2, 47\}, \{13, 6, 27\}, \{12, 40, 26, 31\}$

donde $\boldsymbol{\theta^i(j,G)}$ es una bacteria $\boldsymbol{i}$ de la población en el proceso quimiotáxico $\boldsymbol{j}$ en una generación $\boldsymbol{G}$. En cada '{ }'se encuentra cada una de las 5 comidas (desayuno, intermedio, almuerzo, intermedio y cena) con sus respectivos alimentos.

Para este problema TS-MBFOA no hace uso del buscador local SQP. El pseudocódigo de TS-MBFOA es presentado en el algoritmo 4. Los parámetros de entrada son: número de bacterias $\boldsymbol{S_b}$, límite del ciclo quimiotáxico $\boldsymbol{N_c}$, número de bacterias a reproducir $\boldsymbol{S_r}$, factor de escalamiento $\boldsymbol{\beta}$ para el operador de agrupamiento, $\boldsymbol{R}$ factor de escalamiento para el tamaño de paso, frecuencia de reproducción $\boldsymbol{RepCycle}$ y número de generaciones $\boldsymbol{GMAX}$. Las modificaciones y nuevos mecanismos propuestos son remarcados con letras negritas.

Crear una población inicial de bacterias $\boldsymbol{\theta^i(j,0)}$ $\forall \boldsymbol{i}$, $\boldsymbol{i = 1,\ldots,S_b}$ usando el mecanismo de sesgo y **redondear al entero más proximo cada variable**
Evaluar $\boldsymbol{f(\theta^i(j,0))}$ $\forall \boldsymbol{i}$, $\boldsymbol{i = 1,\ldots,S_b}$
for *G=1 to* $\boldsymbol{GMAX}$ **do**
  for *i=1 to* $\boldsymbol{S_b}$ **do**
    for *j=1 to* $\boldsymbol{N_c}$ **do**
      Perform the **En el proceso quimiotáxico intercalar los nados propuestos** con las Ecs. 7.1 y 7.2. Aplicar el operador de agrupamiento con la Ec. 3.4 usando $\boldsymbol{\beta}$ para la bacteria $\boldsymbol{\theta^i(j,G)}$. **Redondear el valor de las variables al entero más proximo.**
    end
  end
  if *G mod* $\boldsymbol{RepCycle}$ *== 0* **then**
    Realizar el proceso de reproducción ordenando la población de acuerdo a las reglas de factibilidad y eliminar a $\boldsymbol{S_r}$ peores bacterias y duplicar el resto de bacterias $\boldsymbol{S_b - S_r}$.
  end
  Realizar el proceso de eliminación-dispersión eliminando a la peor bacteria $\boldsymbol{\theta^w(j,G)}$ de la población actual considerando la técnica de manejo de restricciones.
  Actualizar el vector de tamaño de paso usando la Ec. 7.3
end

Algoritmo 4: Pseudocódigo de TS-MBFOA para espacio de números enteros.

Interfaz gráfica y funcionamiento del generador de menú nutricional

Una interfaz fue diseñada en Matlab para el generador de menús nutricionales utilizando la Interfaz Gráfica de Usuario (GUI por sus siglas en Ingles). La interfaz es sencilla y muy fácil de entender-utilizar. En la Figura 7.22 se muestra la interfaz del generador de menús.

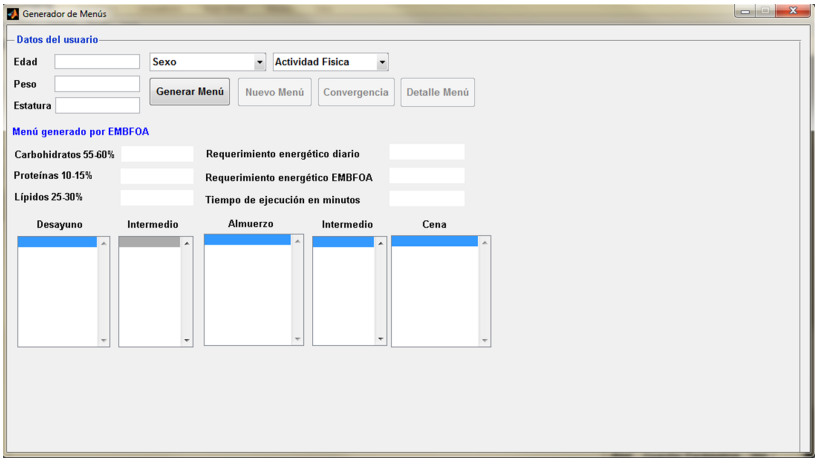

Figura 7.22: Interfaz del generador de menús basado en TS-MBFOA.

Para generar un menú personalizado el usuario debe introducir los datos de entrada los cuales son: edad, peso, estatura, seleccionar el sexo y el tipo de actividad física. Un ejemplo del funcionamiento

del generador de menús se presenta en la Figura 7.23.

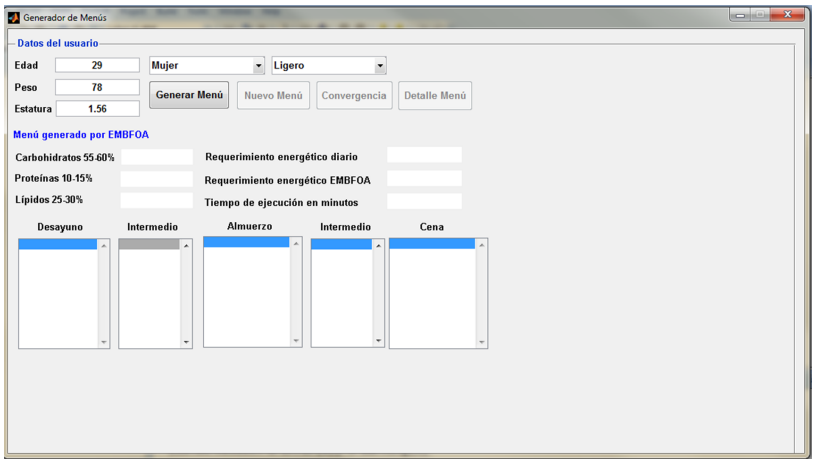

Figura 7.23: Generador de menús con datos de entrada.

Después de introducir los datos de entrada del Individuo, el usuario del generador debe hacer clic en el botón Generar Menú y esperar algunos minutos para observar en pantalla el menú generado por el algoritmo TS-MBFOA. El tiempo de respuesta del generador depende del número de evaluaciones máximas permitidas al algoritmo, las cuales pueden ser ajustadas por el usuario.

El menú generado con los datos de entrada de la Figura 7.23 se muestran en la Figura 7.24. Después de 3.32 minutos, el generador muestra el menú encontrado por TS-MBFOA dividido en las 5 comidas (Desayuno, intermedio, almuerzo, intermedio y cena) el cual debe cumplir con el ***ReqEnergInd*** y los porcentajes de carbohidratos, lípidos y proteínas con un margen de error del 10 % calculados por el generador Con base enl modelado matemático presentado en párrafos anteriores.

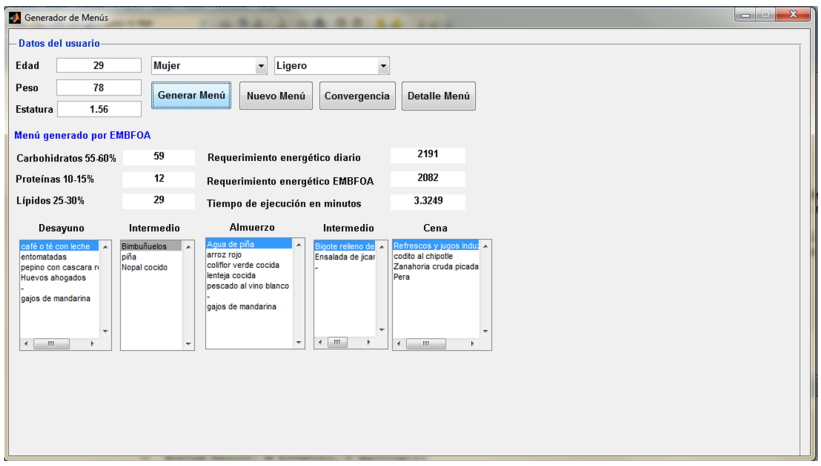

Figura 7.24: Ejemplo de resultado del generador de menús.

Para visualizar los detalles del menú, es decir, visualizar las calorias, lípidos, proteínas y carbohidratos de cada alimento del menú, se debe dar clic en el botón Detalle Menú y el generador mostrará una tabla con el detalle del menú en la parte superior derecha de la pantalla como se muestra en la Figura 7.25. Al final de la tabla, el generador presenta la sumatoria de las calorías, proteínas, lípidos

y carbohidratos totales del menú encontrado por TS-MBFOA y también muestra en la fila final la cantidad de calorías, proteínas, lípidos y carbohidratos exactos que requiere el individuo para que el usuario haga una comparación entre ellos y tome la decisión de tomar ese menú o generar uno nuevo. Para generar un nuevo menú el usuario debe dar clic sobre el botón Nuevo Menú y seguir los pasos de la Figura 7.23.

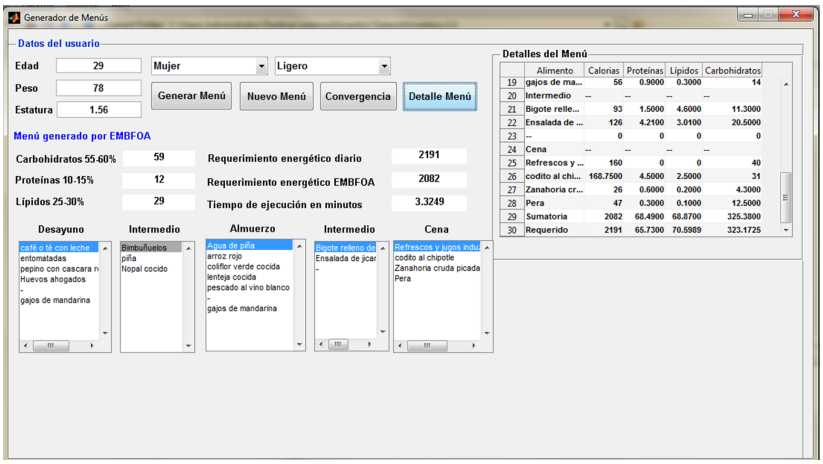

Figura 7.25: Visualización del detalle del menú generado.

Para visualizar el comportamiento del algoritmo y comprobar que el menú no viola ninguna de las restricciones se debe dar clic en el botón Convergencia el cual mostrará una gráfica de convergencia del algoritmo (ver Figura 7.26), el valor final de la función objetivo (valor F. Obj en la pantalla), es decir, el error que hay entre el ***ReqEnergInd*** y el requerimiento energético encontrado por TS-MBFOA y el valor de la Suma de Violación de Restricciones (SVR, V. Restricciones en la pantalla) que es la sumatoria de las restricciones si son violadas. Si el valor de V. Restricciones es 0, el menú generado por TS-MBFOA satisface todas las restricciones del modelado matemático, es decir, el menú cumple con las leyes de la alimentación y demás especificaciones mencionadas. De lo contrario, el menú viola alguna de las restricciones lo cual se puede detectar en las últimas dos filas de la tabla del detalle del menú donde la diferencia entre ambas filas es notoriamente elevada. En esta parte de la interfaz, el usuario puede modificar los parámetros básicos del algoritmo y volver a generar un nuevo menú, para ello debe dar clic en el botón Ajustar parámetros del algoritmo que aparece en la parte inferior derecha de la pantalla y hacer los cambios que considere. Posteriormente debe dar clic en el botón Guardar parámetros que aparece en la misma posición del botón Ajustar parámetros del algoritmo y por último dar clic en el botón Nuevo menú. Los parámetros por default del generador de menús basado en TS-MBFOA se presentan en la Tabla 7.13.

La interfaz del generador de menús también muestra los ingredientes y modo de preparación de los alimentos que tengan dicha información, para lo cual se debe dar clic sobre alguno de los alimentos y en la parte inferior de la pantalla aparecerá la información (Ver Figura 7.27).

La solución de TS-MBFOA para el conjunto de valores de entrada de la Figura 7.23 es: {18, 48, 25, 19, 2, 47}, {17, 43, 23}, {4, 39, 19, 19, 28, 2, 47}, {13, 6, 27}, {12, 40, 26, 31} con la cual se genera el menú resultante de la Figura 7.27 y cada uno de los valores hace referencia a un index en la base de alimentos en uno de los 10 grupo.

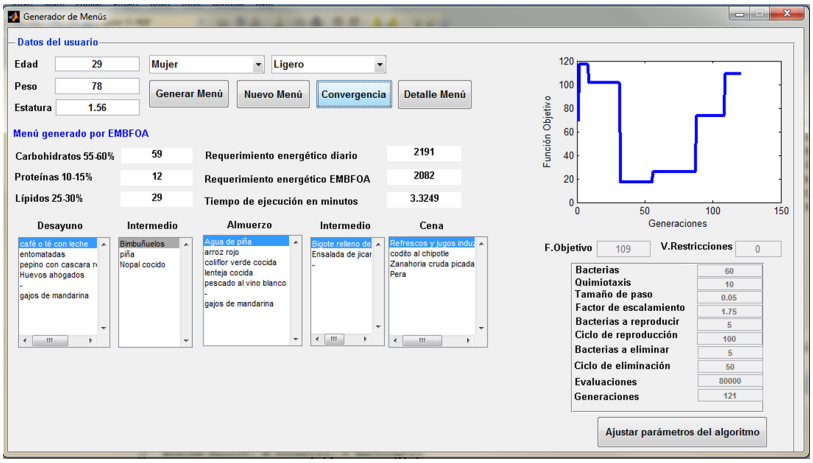

Figura 7.26: Visualización de la convergencia del algoritmo al generar un menú.

Parámetros	**valores**
Bacterias $\boldsymbol{S_b}$	60
Limite del ciclo quimiotáxico $\boldsymbol{N_c}$	10
Tamaño de paso $\boldsymbol{R}$	0.05
Factor de escalamiento $\boldsymbol{\beta}$	1.75
Bacterias a reproducir $\boldsymbol{S_r}$	5
Frecuencia de reproducción $\boldsymbol{RepCyle}$	100
Evaluaciones	80,000

Tabla 7.13: Parámetros de TS-MBFOA en el generador de menús.

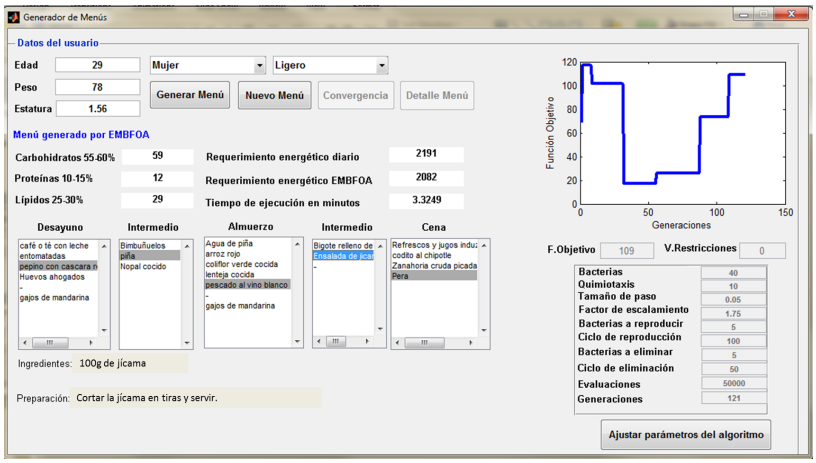

Figura 7.27: Visualización de los ingredientes y modo de preparación de los alimentos.

Resultados del generador de menús

El generador de menús fue probado en 15 individuos diferentes, la información de cada individuo se presenta en la Tabla 7.14.El generador se ejecutó solo una vez por cada individuo con los valores por default en los parámetros para TS-MBFOA presentados en la Tabla 7.13. El resultado del generador de menús para cada uno de los 15 se presenta en la Tabla 7.15.

Individuo	Edad	Peso (kg)	Estatura (mtrs)	Sexo	Nivel de actividad
1	21	60.5	1.46	Femenino	Moderado
2	21	98	1.68	Femenino	Ligero
3	28	78	1.56	Femenino	Moderado
4	24	61.5	1.52	Femenino	Intenso
5	23	63.5	1.63	Masculino	Moderado
6	22	69	1.59	Femenino	Intenso
7	19	119	1.72	Masculino	Ligero
8	20	67	1.54	Femenino	Moderado
9	20	61.5	1.59	Femenino	Intenso
10	23	46.5	1.51	Femenino	Moderado
11	20	55.5	1.57	Femenino	Moderado
12	22	61.5	1.53	Femenino	Ligero
13	22	84	1.75	Masculino	Intenso
14	22	64	1.54	Femenino	Intenso
15	21	51	1.53	Femenino	Ligero

Tabla 7.14: Información de 15 individuos para probar el generador de menús.

De acuerdo a los resultados de la Tabla 7.15 TS-MBFOA encontró menús que satisfacen la restricciones del problema en 4 de los 15 individuos. Para el resto de los individuos, TS-MBFOA encontró menús con una mínima suma de violación de restricciones. El tiempo requerido por el generador de menús fue diferentes en los 15 casos pero en promedio el tiempo de ejecución es 2.2 minutos. Es importante mencionar que el generador de menús solo ejecuta una vez a TS-MBFOA en cada caso y la base de alimentos tiene un número básico de alimentos por cada grupo. Sin embargo, TS-MBFOA fue capaz de encontrar menús variados con la mínima suma de violación de retricciones. La Figura 7.28 muestra el menú generado por TS-MBFOA con los datos de los individuos 3.

Individuo	Lípidos	Proteínas	Carbohidrátos	ReqEnergInd	TS-MBFOA	F. Obj	SVR	Minutos
1	29	12	59	1892	1852	40	1.46	3.3
2	28	14	58	2328	2298.25	29.75	2.95	2.1
3	29	12	59	2199	2161.3	37.75	0	3.4
4	29	12	59	2189	2131.5	57.5	2.36	3
5	29	12	59	2333	2293.8	39.25	0	1.8
6	28	13	59	2407	2257.8	149.25	15.96	2.3
7	28	13	59	2871	2772.8	98.25	5.65	1.8
8	28	13	59	2073	2003.3	69.75	0.024	1.9
9	28	14	58	2296	2215	81	0.64	2.1
10	28	13	59	1712	1644	68	6.86	1.8
11	29	14	57	1928	1888	40	0	1.8
12	29	14	57	1716	1724	8	1.33	1.8
13	27	14	59	3468	3375	93	11.35	2.1
14	29	13	58	2525	2471.8	53.25	0	1.9
15	28	13	59	1584	1563	21	5.83	2.6

Tabla 7.15: Resultado del generador de menús usando TS-MBFOA.

En general, TS-MBFOA es un algoritmo capaz de encontrar soluciones factibles altamente competitivas en problemas del mundo real como los tres problemas de diseño Mecatrónico derivados de la Síntesis de un mecanismo plano de cuatro barras y la generación de menús nutricionales. Aunque ambos casos son diferentes en área y en representación numérica, TS-MBFOA puede ser implementado sin sufrir cambios severos y obtener resultados competitivos, en el caso de la generación de menús nutricionales, TS-MBFOA solo fue adaptado para convertir el valor de cada variable de decisión en el entero más próximo.

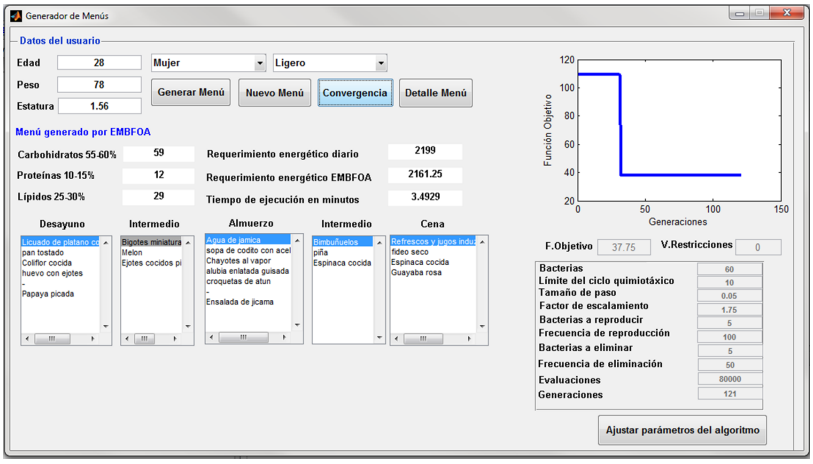

Figura 7.28: Resultado de TS-MBFOA con los datos del individuo 3.

Capítulo 8

Conclusiones y trabajos futuros

En este trabajo el algoritmo basado en el forrajeo de bacterias modificado (MBFOA por sus siglas en Ingles) fue adaptado para mejorar su rendimiento en problemas de optimización global con restricciones. Diversas propuestas fueron derivadas, experimentadas y probadas en problemas de pruebas y del mundo real con el fin de crear un algoritmo competitivo y robusto.

En un primer experimento se midió el rendimiento de MBFOA usando un mecanismo de tolerancia dinámica para funciones con presencia de restricciones de igualdad en las primeras generaciones del algoritmo y cinco diferentes técnicas para el manejo de restricciones altamente utilizadas en el estado del arte, las cuales son: 1) las Reglas de Factibilidad de Deb, 2) Función de Penalización, 3) Penalización de Muerte, 4) Penalización Adaptativa y ϵ-Constraint, como resultado, las reglas de factibilidad son incluidas a MBFOA por favorecer el rendimiento del algoritmo más que cualquier otra técnica.

De acuerdo a la literatura especializada, el tamaño de paso es el parámetro más sensible del algoritmo basado en el forrajeo de bacterias, en un segundo experimento se probó a MBFOA con cuatro mecanismos diferentes de control para el tamaño de paso los cuales fueron: 1) dinámico, 2) adaptativo, 3) estático y 4) aleatorio. Además, la presencia del proceso de reproducción fue controlada para evitar convergencia prematura. Los resultados no presentaron una diferencia significativa entre las variantes de MBFOA con tamaño de paso aleatorio, dinámico y adaptativo las cuales obtivieron mejores resultados que la versión estática.

En un tercer experimento se propuso usar el método secuencial de programación cuadrática como buscador local en ciertas generaciones de MBFOA usando un tamaño de paso dinámico. En esta propuesta el mecanismo de tolerancia dinámica para restricciones de igualdad no fue necesario. Además dos operadores de nados fueron intercalados en el proceso quimiotáxico con el fin de explorar y explotar el espacio de búsqueda, uno es el nado original donde la bacteria avanza un tamaño de paso en una dirección de búsqueda y en el segundo nado la bacteria toma como tamaño de paso la dirección de búsqueda. Los resultados fueron competitivos a otros algoritmos del estado del arte. Sin embargo, en problemas altamente restringidos y de alta dimensionalidad el algoritmo careció de robustez. La presencia de un buscador local benefició a MBFOA, sin embargo, el algoritmo fue capaz de generar soluciones factibles desde las primeras generaciones, por lo tanto, SQP solo fue utilizado dos veces en las iteraciones del algoritmo. La convergencia prematura fue el principal problema del algoritmo derivado.

En el último experimento, se buscó evitar la convergencia prematura y favorecer la diversidad de la población. Dos nados fueron intercalados en el proceso quimiotáxico: el nado original con tamaño

de paso aleatorio y el segundo nado es el operador de mutación usado en los algoritmos evolutivos con el fin de mejorar la capacidad de exploración y explotación del algoritmo. Además, el proceso de eliminación-dispersión tiene lugar en ciertos ciclos del proceso de búsqueda con mayor número de bacterias. SQP sigue presente en la propuesta en dos ocasiones durante el proceso de búsqueda. Por último, motivado por el creciente interés en las técnicas de inicialización de algoritmos inspirados en la naturaleza, un mecanismo de inclinación para crear la población de las bacterias fue propuesto, la población inicial de bacterias está integrado por tres grupos: uno con bacterias aleatoria sesgadas al límite inferior de las variables de decisión, otro con bacterias aleatorias sesgadas al límite superior de las variables de decisión y un grupo de bacterias aleatorias sin sesgo.

La propuesta derivada fue llamada Evolutivo MBFOA (EMBFOA) y aplicada para resolver problemas de prueba, tres casos de estudios derivados de la síntesis de un mecanismo plano de cuatro barras y para generar menús nutricionales. En todos los problemas el algoritmo obtuvo resultados altamente competitivos al compararse con otros algoritmos del estado del arte y en algunos casos, con un número menor de evaluaciones. Para la generación de menús nutricionales, EMBFOA solo fue adaptado para convertir el valor de cada variable de decisión en el entero más próximo.

EMBFOA fue probado en problemas de distintas áreas y de diferente representación numérica con resultados exitosos demostrando su capacidad de generar soluciones factibles altamente competitivas tanto en problemas de prueba como del mundo real. Sin embargo, trabajos futuros quedan por realizar, como la comparación de los menús generados por el algoritmo propuesto contra menús generados en sistemas expertos no comerciales, además de obtener el punto de vista de un nutriólogo para validar los menús. Por otra parte, está la revisión de los dos operadores de nados con el propósito de disminuir el número de evaluaciones requeridas por el algoritmo para obtener resultados competitivos en problemas del mundo real y se estudiarán distintas técnicas de normalización para ser aplicadas al espacio de estos problemas.

Referencias

[1] Patrick Siarry and Zbigniew Michalewicz, editors. *Advances in Metaheuristic Methods for Hard Optimization*. Springer, Berlin, 2008. ISBN 978-3-540-72959-4.

[2] A. Eiben and J. E. Smith, editors. *Introduction to Evolutionary Computing*. Natural Computing Series. Springer-Verlag, 2003.

[3] Andries P. Engelbrecht. *Fundamentals of Computational Swarm Intelligence*. John Wiley & Sons, 2005.

[4] James Kennedy and Russell C. Eberhart. *Swarm Intelligence*. Morgan Kaufmann, UK, 2001.

[5] Xiaohui Hu and Russell Eberhart. Solving Constrained Nonlinear Optimization Problems with Particle Swarm Optimization. In *Proceedings of the 6th World Multiconference on Systemics, Cybernetics and Informatics (SCI 2002)*, volume 5. Orlando, USA, IIIS, July 2002.

[6] Xiaohui Hu, Russell C. Eberhart, and Yuhui Shi. Engineering Optimization with Particle Swarm. In *Proceedings of the 2003 IEEE Swarm Intelligence Symposium*, pages 53–57. Indianapolis, Indiana, USA, IEEE Service Center, April 2003.

[7] K.E. Parsopoulos and M.N. Vrahatis. Unified Particle Swarm Optimization for solving constrained engineering optimization problems. *Advances in Natural Computation, Pt. 3*, pages 582–591, 2005. Lecture Notes in Computer Science Vol. 3612.

[8] Leticia Cagnina, Susana Esquivel, and Carlos Coello-Coello. A Bi-population PSO with a Shake-Mechanism for Solving Constrained Numerical Optimization. In *2007 IEEE Congress on Evolutionary Computation (CEC'2007)*, pages 670–676, Singapore, September 2007. IEEE Press.

[9] Efren Mezura-Montes and Jorge Isacc Flores-Mendoza. Improved particle swarm optimization in constrained numerical search spaces. In Raymond Chiong, editor, *Nature-Inspired Algorithms for Optimization*, volume 193, pages 299–332. Springer-Verlag, Studies in Computational Intelligence Series, 2009, ISBN: 978-3-540-72963-1., 2009.

[10] Hans J. Bremermann. Chemotaxis and optimization. *J. Franklin Inst*, 297:397–404, 1974.

[11] Kevin M. Passino. Biomimicry of bacterial foraging for distributed optimization and control. *IEEE Control Systems Magazine*, 22(3):52–67, 2002.

[12] H. Alvarez, O. Quintero, A. Angel, and M. Yepes. Strategies for avoiding local extreme in function optimization through bacterial chemotaxis. In *Memorias del X Congreso Latinoamericano de Control Automático (CLCA 2002)*. Guadalajara, México, Diciembre 2002.

[13] W.J.Tang, Q.H.Wu, and J.R.Saunders. A bacterial swarm algorithm for global optimization. In *Proceedings of the IEEE Congress on Evolutionary Computation (CEC'2007)*, pages 1207–1212. Singapore, IEEE Service Center, September 2007.

[14] Ying Chu, Hua Mi, Huilian Liao, Zhen Ji, and Q. H. Wu. A fast bacterial swarming algorithm for high-dimensional function optimization. In *Congress on Evolutionary Computation (CEC 2008)*, pages 3134–3139, 2008.

[15] Sambarta Dasgupta, Arijit Biswas, Swagatam Das, Bijaya Ketan Panigrahi, and Ajith Abraham. A micro-bacterial foraging algorithm for high-dimensional optimization. In *Congress on Evolutionary Computation (CEC 2009)*, pages 3134–3139, 2009.

[16] S. D. Muller, Jarno M., S. A., and Petros K. Optimization based on bacterial chemotaxis. *IEEE Transactions on Evolutionary Computation*, 6(1):16–29, 2002.

[17] K.M.Bakwad, S.S.Pattniak, B.S.Sohi, S.Devi, S.Gollapudi, C.Sagar, and P.K Patra. Synchronous bacterial foraging optimization for multimodal and high dimensional functions. In *Proceedings of the 2008 International Conference on Computing, Communication and Networking (ICCCN 2008)*, pages 1–8, 2008.

[18] Ritanjali Majhi, Ganapti Panda, Gadadhar Sahoo, Pradipta K. Dash, and D. P. Das. Stock market prediction of s&p 500 and djia using bacterial foraging optimization technique. In *Proceedings of the IEEE Congress on Evolutionary Computation (CEC'2007)*, pages 2569–2575. Singapore, IEEE Service Center, September 2007.

[19] Vitorino Ramos, C. Fernandes, A. C. Rosa, and A.Abraham. Computational chemotaxis in ants and bacteria over dynamic environments. In *Proceedings of the IEEE Congress on Evolutionary Computation (CEC'2007)*, pages 1109–1117. Singapore, IEEE Service Center, September 2007.

[20] Jae-Hoon Cho, Myung-Geun Chun, and Dae-Jong Lee. Parameter optimization of extreme learning machine using bacterial foraging algorithm. In *Proceedings of the 8th Symposium on Advanced Intelligent systems (ISIS 2007)*, pages 742–747, 2007.

[21] Babita Majhi and Ganapti Panda. Bacteria foraging based identificacion of nonlinear dynamic system. In *Proceedings of the IEEE Congress on Evolutionary Computation (CEC'2007)*, pages 1636–1641. Singapore, IEEE Service Center, September 2007.

[22] Z.Lu, M.S.Li, Z. Lu, M. S. Li, W. J. Tang, and Q. H. Wu. Optimal location of FACTS devices by a bacterial swarming algorithm for reactive power planning. In *Proceedings of the IEEE Congress on Evolutionary Computation (CEC'2007)*, pages 2344–2349. Singapore, IEEE Service Center, September 2007.

[23] B. Sumanbabu, S. Mishra, B.K.Panigrahi, and G.K.Venayagamoorthy. Robust tuning of modern power systems stabilizers using bacterial foraging algorithm. In *Proceedings of the IEEE Congress on Evolutionary Computation (CEC'2007)*, pages 2317–2324. Singapore, IEEE Service Center, September 2007.

[24] K.M. Bakwad, S.S.Pattnaik, B.S.Sohi, S.Devi, B.K.Panigrahi, and S.V.Gollapudi. Bacterial foraging optimization technique cascaded with adaptive filter to enhance peak signal to noise ratio from single image. *IETE J RES*, 1(55):173–179, 2009.

[25] Swagatam Das, Archana Chowdhury, and Ajith Abraham. A bacterial evolutionary algorithm for automatic data clustering. In *Congress on Evolutionary Computation (CEC 2009)*, pages 2403–2410, 2009.

[26] K. M. Bakwad, S.S. Pattnaik, B. S. Sohi, S. Devi, and M.R. Lohakare. Parallel bacterial foraging optimization for video compression. *Engineering Optimization*, 1(1):118–122, 2009.

[27] Arijit Biswas, Swagatam Das, Ajith Abraham, and Sambarta Dasgupta. Analysis of the reproduction operator in an artificial bacterial foraging system. *Applied Mathematics and Computation*, 0(215):3343–3355, 2010.

[28] Swagatam Das, Sambarta Dasgupta, Arijit Biswas, and Tribeni Prasad Banerjee. On stability of the chemotactic dynamics in bacterial foraging optimization algorithm. In *Proccedings in 2nd National Conference Mathematical Techniques: Emerging Paradigms for Electronics and IT Industries (MATEIT-2008)*, pages 153–158, 2008.

[29] Sambarta Dasgupta, Swagatam Das, Ajith Abraham, and Arijit Biswas. Adaptive computational chemotaxis in bacterial foraging optimization: An analysis. *IEEE Transactions on Evolutionary Computation*, 13(4):919–941, 2009.

[30] Dong Hwa Kim, Ajith Abraham, and Jae Hoon Cho. A hybrid genetic algorithm and bacterial foraging approach for global optimization. *Information Sciences*, 177(18):3918–3937, 2007.

[31] Arijit Biswas, Sambarta Dasgupta, Swagatam Das, and Ajith Abraham. Synergy of PSO and bacterial foraging optimization - a comparative study on numerical benchmarks. In E. Corchado et al., editor, *Innovations in Hybrid Intelligent Systems 2007*, pages 255–263. Springer-Verlag, 2007.

[32] Wael Korani. Bacterial foraging oriented by particle swarm optimization strategy for pid tuning. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2008)*, pages 1823–1826, Atlanta, GA, USA, 2008. ISBN:978-1-60558-131-6.

[33] Arijit Biswas, Sambarta Dasgupta, Swagatam Das, and Ajith Abraham. A Synergy of Differential Evolution and Bacterial Foraging Optimization for global optimization. *Neural Network World*, 17:607–626, 2007.

[34] Efrén Mezura-Montes and Elyar A. López-Davila. Adaptation and local search in the modified bacterial foraging algorithm for constrained optimization. In *Proceedings of the IEEE Congress on Evolutionary Computation 2012*, pages 497–504. ISBN:978-1-4673-1508-1, 2012.

[35] Alireza Kasaiezadeh, Amir Khajepour, and Steven L. Waslander. Spiral bacterial foraging optimization method: Algorithm, evaluation and convergence analysis. *Engineering Optimization*, 46(4):439–464, 2014.

[36] Efrén Mezura-Montes and Betania Hernández-Ocaña. Modified bacterial foraging optimization for engineering design. In *Proceedings of the Artificial Neural Networks in Enginnering Conference (ANNIE'2009)*, volume 19, pages 357–364. in Cihan H. Dagli et al. (editors), ASME Press Series, Intelligent Engineering Systems Through Artificial Neural Networks, 2009.

[37] Kalyanmoy Deb. An Efficient Constraint Handling Method for Genetic Algorithms. *Computer Methods in Applied Mechanics and Engineering*, 186(2-4):311–338, 2000.

[38] Efrén Mezura-Montes, Edgar Alfredo Portilla-Flores, and Betania Hernández-Ocaña. Optimization of a mechanical design problem with the modified bacterial foraging algorithm. In *Proceedings of the XVII Argentine Congress on Computer Sciences*, La Plata, Argentina, October 2011.

[39] Betania Hernández-Ocaña, Efrén Mezura-Montes, and Pilar Pozos-Parra. A review of the bacterial foraging algorithm in constrained numerical optimization. In *Proceedings of the Congress on Evolutionary Computation (CEC'2013)*, pages 2695–2702. IEEE, 2013.

[40] Efrén Mezura-Montes and Carlos A. Coello Coello. Constraint-handling in nature-inspired numerical optimization: Past, present and future. *Swarm and Evolutionary Computation*, 1(4):173–194, 2011.

[41] Alice E. Smith and David W. Coit. Constraint Handling Techniques—Penalty Functions. In Thomas Bäck, David B. Fogel, and Zbigniew Michalewicz, editors, *Handbook of Evolutionary Computation*, pages C5.2–1–C5.2–6. Oxford University Press and Institute of Physics Publishing, 1997.

[42] K.M. Miettinen. *Nonlinear Multiobjective Optimization*. Kluwer Academic Publishers, Boston Massachusetts, 1999.

[43] Amitava Chatterjee and Fumitoshi Matsuno. Bacterial foraging techniques for solving ekf-based slam problems. In *Proc. International Control Conference (Control2006)*, Glasgow,UK, Aug. 30-Sep.01 2006.

[44] Dong Hwa Kim and Jae Hoon Cho. Advanced bacterial foraging and its application using fuzzy logic based variable step size and clonal selection of immune algorithm. In *Proceedings of the 6 International Conference on Hybrid Information Technology*, pages 293–298. IEEE, 2006.

[45] A. Biswas, S. Dasgupta, S. Das, and A. Abraham. A synergy of differential evolution and bacterial foraging optimization for global optimization. *Neural Network World*, 17(6):607–626, 2007.

[46] D.P. Acharya, G. Panda, S. Mishra, and Y.V.S. Lakshmi. Bacterial foraging based independent component analysi. In *International Conference on Computational Intelligence and Multimedia Applications*, pages 527–531. IEEE, 2007.

[47] B.Sumanbabu, S.Mishra, B.K.Panigrahi, and G.K.Venayagamoorthy. Robust tuning of modern power systems stabilizers using bacterial foraging algorithm. In *Proceedings of the IEEE Congress on Evolutionary Computation (CEC'2007)*, pages 2317–2324. Singapore, IEEE Service Center, September 2007.

[48] M. Hanmandlu, A.V. Nath, A.C. Misrha, and V.K. Madasu. Fuzzy model based recognition of handwritten hindi numerals using bacterial foraging. In *6th International Conference on Computer and Information Science (ICIS 2007)*, pages 309–314. IEEE, 2007.

[49] T. Datta and I.S. Misra. Improved adaptive bacteria foraging algorithm in optimization of antenna array for faster convergence. *Electromagnetics Research*, 1:143–157, 2008.

[50] Madhubanti Maitra and Amitava Chatterjee. A novel technique for multilevel optimal magnetic resonance brain image thresholding using bacterial foraging. *Measurement*, 0(41):1124–1134, 2008.

[51] Hanning Chen, Yunlong Zhu, and Kunyuan Hu. Cooperative bacterial foraging optimization. *Hindawi: Discrete Dynamics in Nature and Society*, 2009:1–17, 2009.

[52] K. Vaisakh, P. Praveena, and S. Rama Mohana Rao. Pso-dv and bacterial foraging optimization based dynamic economic dispatch with non-smooth cost functions. In *International Conference on Advances in Computing, Control, and Telecommunication Technologies*, pages 135–139. IEEE, 2009.

[53] Hsiang-Cheh Huang, Yueh-Hong Chen, and Ajith Abraham. Optimized watermarking using swarm-based bacterial foraging. *Information Hiding and Multimedia Signal Processing*, 1(1):51–58, 2010.

[54] Yudong Zhang, Lenan Wu, and Shuihua Wang. Bacterial foraging optimization based neural network for short-term load forecasting. *Computational Information Systems*, 6(7):2099–2105, 2010.

[55] John Oyekan and Huosheng Hu. A novel bacterial foraging algorithm for automated tuning of pid controllers of uavs. In *Proccedings of the International Conference on Information and Automation*, pages 693–698, Harbin China, 2010. IEEE.

[56] Amitava Chatterjee, Mourad Fakhfakh, and Patrick Siarry. Design of second-generation current conveyors employing bacterial foraging optimization. *Microelectronics Journal*, 0(41):616–626, 2012.

[57] K. Sathish Kumar and T. Jayabarathi. Power system reconfiguration and loss minimization for an distribution systems using bacterial foraging optimization algorithm. *Electrical Power and Energy Systems*, 0(36):13–17, 2012.

[58] Youxin Luo and Zhaoguo Chen. Optimization for pid control parameters on hydraulic servo control system based on the novel compound evolutionary algorithm. In *Second International Conference on Computer Modeling and Simulation*, pages 40–43. IEEE, 2010.

[59] Mario A. Muñoz, Saman K. Halgamuge, Wilfredo Alfonso, and Eduardo F. Caicedo. Simplifying the bacteria foraging optimization algorithm. In *IEEE Congress on Evolutionary Computation (CEC)*, pages 1–7, Australia, 2010.

[60] H. Supriyono and M.O. Tokhi. Bacterial foraging algorithm with adaptable chemotactic step size. In *Proccedings of the Second International Conference on Computational Intelligence, Communication Systems and Networks*, pages 72–77, Liverpool, United Kingdom, 2010. IEEE.

[61] Ben Niu, Hong Wang, Lijing Tan, and Li Li. Improved bfo with adaptive chemotaxis step for global optimization. In *In proceedings of Seventh International Conference on Computational Intelligence and Security*, pages 76–80, Hainan, 2011. ISBN:978-1-4577-2008-6.

[62] Hanning Chen, Yunlong Zhu, Kunyuan Hu, and Tao Ku. Dynamic rfid network optimization using a self-adaptive bacterial foraging algorithm. *Artificial Intelligence*, 7(11):219–231, 2011.

[63] S.M. Tabatabaei and B. Vahidi. Bacterial foraging solution based fuzzy logic decision for optimal capacitor allocation in radial distribution system. *Electrical Power Systems Research*, 0(81):1045–1050, 2011.

[64] Nandita Sanyal, Amitava Chatterjee, and Sugata Munshi. An adaptive bacterial foraging algorithm for fuzzy entrophy based image segmentation. *Expert Systems with Applications*, 0(38):15489–15498, 2011.

[65] P. D. Sathya and R. Kayalvizhi. Optimal multilevel thresolding using bacterial foraging algorithm. *Expert Systems with Applications*, 0(38):15549–15564, 2011.

[66] P. D. Sathya and R. Kayalvizhi. Optimal segmentation of brain mri based on adaptive bacterial foraging algirithm. *Neurocomputing*, 0(74):2299–2313, 2011.

[67] P. D. Sathya and R. Kayalvizhi. Modified bacterial foraging based multilevel thresholding for image segmentation. *Engineering Applications of Artificial Intelligence*, 0(24):595–615, 2011.

[68] Sandeep Sharma and B.K. Kanaujia. Optimization of resonant frequency of circular microstrip antenna with and without air gaps using bacterial foraging optimization technique. In *Proccedings of the International Conference on Computational Intelligence and Communication Systems*, pages 574–577. IEEE, 2011.

[69] Xiujuan Lei, Shuang Wu, Liang Ge, and Aidong Zhang. Clustering ppi data based on bacterial foraging optimization algorithm. In *International Conference on Bioinformatics and Biomedicine*, pages 96–99. IEEE, 2011.

[70] Om Prakash Verma, Madasu Hanmandlu, Puneet Kumar, Sidharth Chhabra, and Akhil Jindal. A novel bacterial foraging technique for edge detection. *Pattern Recognition Letters*, 0(32):1187–1196, 2011.

[71] Susheree Sangita Patnaik and Anup Kumar Panda. Particle swarm optimization and bacterial foraging optimization techniques for optimal current harmonic mitigation by employing active power filter. *Hindawi: Appied Computational Intelligence and Soft Computing*, 2012:1–10, 2012.

[72] Shima Kamyab and Abbas Bahrololoum. Designing of rule base for a tsk-fuzzy system using bacterial foraging optimization algorithm (bfoa). *Social and Behavioral Sciences*, 0(32):176–183, 2012.

[73] H. Supriyono and M.O. Tokhi. Dynamic neuro-modelling using bacterial foraging optimisation with fuzzy adaptation. In *Proccedings of the Third International Conference on Intelligence Systems Modelling and Simulation*, pages 109–114. IEEE, 2012.

[74] S.M. Abd-Elazim and E.S. Ali. Bacterial foraging optimization algorithm based svc damping controller design for power system stability enhancement. *Electrical Power and Energy Systems*, 0(43):933–940, 2012.

[75] K. Vivekanandan and D. Ramyachitra. Bacteria foraging optimization for protein sequence analysis on the grid. *Future Generation Computer Systems*, 0(28):647–656, 2012.

[76] Hossein Nouri and Tang Sai Hong. A bacterial foraging algorithm based cell information considering operation time. *Manufacturing Systems*, (31):326–336, 2012.

[77] Nikhil Kushwaha, Vimal Singh Bisht, and Gautam Shah. Genetic algorithm based bacterial foraging approach for optimization. *IJCA Proceedings on National Conference on Future Aspects of Artificial intelligence in Industrial Automation 2012*, NCFAAIIA(2):11–14, May 2012.

[78] Efrén Mezura-Montes and Betania Hernández-Ocaña. Modified bacterial foraging optimization for engineering design. In Cihan H. Dagli and et al., editors, *Proceedings of the Artificial Neural Networks in Engineering Conference (ANNIE'2009)*, volume 19 of *Intelligent Engineering Systems Through Artificial Neural Networks*, pages 357–364, St. Louis, MO, USA, November 2009. ASME Press.

[79] P. Praveena, K. Vaisakh, and S. Rama Mohana Rao. A bacterial foraging and PSO-DE algorithm for solving dynamic economic dispatch problem with valve-point effects. In *First International Conference on Integrated Intelligent Computing*, pages 227–232. IEEE, 2010.

[80] Chang Chunguang, Zhu Yunlong, Hu Kunyuan, and Shen Hai. Research on smelting ingredient diluting for refined copper by bacterial foraging optimization algorithm. In *International Conference on Digital Manufacturing and Automation*, pages 275–278. IEEE, 2010.

[81] Kou Wei, Sun Feng-rui, Yang Li, and Chen Lin-gen. Application of improved bcc algorithm and rbfnn in identification of defect parameters. In *Fourth International Conference on Genetic and Evolutionary Computing*, pages 160–164. IEEE, 2010.

[82] B.K. Panigrahi, V. Ravikumar Pandi, Sanjoy Das, and Swagatam Das. Multiobjective fuzzy dominance based bacterial foraging algorithm to solve economic emmission dispatch dispatch problem. *Energy*, 35(12):4761–4770, 2010.

[83] B.K. Panigrahi, V. Ravikumar Pandi, Renu Sharma, Swagatam Das, and Sanjoy Das. Multi-objetive bacteria foraging algorithm for electrical load dispatch problem. *Energy Conversion and Management*, 52(2):1334–1342, 2011.

[84] Paras Deshpande, Deepak Shukla, and M.K. Tiwari. Fuzzy goal programming for inventory management: A bacterial foraging approach. *European Journal of Operational Research*, 0(212):325–336, 2011.

[85] Nicole Pandit, Anshul Tripathi, Shashikala Tapaswi, and Manjaree Pandit. An improved bacterial foraging algorithm for combined static/dynamic environmental economic dispatch. *Applied Soft Computing*, (12):3500–3513, 2012.

[86] K. Vaisakh, P. Praveena, S. Rama Mohana Rao, and Kala Meah. Solving dynamic economic dispatch problem with security constraints using bacterial foraging PSO-DE algorithm. *Electrical Power and Energy Systems*, (39):56–67, 2012.

[87] Ahmed Yousuf Saber. Economic dispatch using particle swarm optimization with bacterial foraging effect. *Electrical Power and Energy Systems*, 0(34):38–46, 2012.

[88] Ben Niu, Yan Fan, Han Xiao, and Bing Xue. Bacterial foraging based approaches to portfolio optimization with liquidity risk. *Neurocomputing*, 98(0):90 – 100, 2012.

[89] Efrén Mezura-Montes, Edgar Alfredo Portilla-Flores, and Betania Hernández-Ocaña. Optimum synthesis of a four-bar mechanism using the modified bacterial foraging algorithm. *International Journal of Systems Science*, 5(45):1080,1100, 2013.

[90] Om Prakash Verma, Rohan Garg, and Vikram Singh Bisht. Optimal time-table generation by hybridized bacterial foraging and genetic algorithms. In *International Conference on Communication Systems and Network Technologies*, pages 919–923. IEEE, 2012.

[91] Linlin Shao and Yuehui Chen. Bacterial foraging optimization algorithm integrating tabu search for motif discovery. In *International Conference on Bioinformatics and Biomedicine*, pages 415–418. IEEE, 2009.

[92] Linlin Shao, Yuehui Chen, and Ajith Abraham. Motif discovery using evolutionary algorithms. In *International Conference of Soft Computing and Pattern Recognition*, pages 420–425. IEEE, 2009.

[93] Efrén Mezura-Montes, editor. *Constraint-Handling in Evolutionary Optimization*, volume 198 of *Studies in Computational Intelligence*. Springer-Verlag, 2009.

[94] Zbigniew Michalewicz and Marc Schoenauer. Evolutionary Algorithms for Constrained Parameter Optimization Problems. *Evolutionary Computation*, 4(1):1–32, 1996.

[95] Slawomir Koziel and Zbigniew Michalewicz. Evolutionary Algorithms, Homomorphous Mappings, and Constrained Parameter Optimization. *Evolutionary Computation*, 7(1):19–44, 1999.

[96] Zbigniew Michalewicz and G. Nazhiyath. Genocop III: A co-evolutionary algorithm for numerical optimization with nonlinear constraints. In David B. Fogel, editor, *Proceedings of the Second IEEE International Conference on Evolutionary Computation*, pages 647–651, Piscataway, New Jersey, 1995. IEEE Press.

[97] Marc Schoenauer and Spyros Xanthakis. Constrained GA Optimization. In Stephanie Forrest, editor, *Proceedings of the Fifth International Conference on Genetic Algorithms (ICGA-93)*, pages 573–580, San Mateo, California, July 1993. University of Illinois at Urbana-Champaign, Morgan Kauffman Publishers.

[98] Thomas P. Runarsson and Xin Yao. Stochastic Ranking for Constrained Evolutionary Optimization. *IEEE Transactions on Evolutionary Computation*, 4(3):284–294, September 2000.

[99] Tetsuyuki Takahama and Setsuko Sakai. Constrained Optimization by the ϵ Constrained Differential Evolution with Gradient-Based Mutation and Feasible Elites. In *2006 IEEE Congress on Evolutionary Computation (CEC'2006)*, pages 308–315, Vancouver, BC, Canada, July 2006. IEEE.

[100] Biruk Tessema and Gary G. Yen. An adaptive penalty formulation for constrained evolutionary optimization. *IEEE Transactions on Systems, Man and Cybernetics Part A—Systems and Humans*, 39(3), 2009.

[101] Guillermo Leguizamón and Carlos A. Coello Coello. Boundary Search for Constrained Numerical Optimization Problems with an Algorithm Inspired on the Ant Colony Metaphor. *IEEE Transactions on Evolutionary Computation*, 13(2):350–368, 2009.

[102] Efren Mezura-Montes and Carlos A. Coello-Coello. Constrained optimization via multiobjective evolutionary algorithms. In David Corne in Joshua Knowles and Kalyanmoy Deb, editors, *Multiobjective Problems Solving from Nature: From Concepts to Applications*, pages 53–76. Springer-Verlag, Natural Computing Series, 2008, ISBN: 978-3-540-72963-1., 2008.

[103] Rammohan Mallipeddi and Ponnuthurai N. Suganthan. Ensemble of Constraint Handling Techniques. *IEEE Transactions On Evolutionary Computation*, 14(4):561–579, August 2010.

[104] Biruk Tessema and Gary G. Yen. An adaptive penalty formulation for constrained evolutionary optimization. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, 39(3):565–578, 2009.

[105] Sana Ben Hamida and Marc Schoenauer. Aschea:new result using adaptive segregational constraint handling. In *Proceedings of the Congress on Evolutionary Computation (CEC'2002)*, pages 884–889, Piscataway, New Jersey, 2002. IEEE.

[106] J.J. Liang, Thomas Philip Runarsson, Efrén Mezura-Montes, Maurice Clerc, P.N. Suganthan, Carlos A. Coello Coello, and K. Deb. Problem definitions and evaluation criteria for the CEC 2006 special session on constrained real-parameter optimization. Technical report, School of EEE Nanyang Technological University, Singapore, September 2006.

[107] María Margarita Reyes Sierra. *Use of Coevolution and Fitness Inheritance for Multi-Objective Particle Swarm Optimization*. PhD thesis, Centro de Investigación y de Estudios Avanzados del Instituto Politécnico Nacional, Ciudad de México, México, Agosto 2006.

[108] Sancho Salcedo-Sanz. A survey of repair methods used as constraint handling techniques in evolutionary algorithms. *Computer Science Review*, 3(4):175–192, 2009.

[109] S.M. Elsayed, R. Sarker, and D. Essam. Integrated strategies differential evolution algorithm with a local search for constrained optimization. In *2011 IEEE Congress on Evolutionary Computation (CEC'2011)*, pages 2618–2625, New Orleans, USA, July 2011. IEEE Press.

[110] Borhan Kazimipour, Xiaodong Li, and AK. Qin. A review of population initialization techniques for evolutionary algorithms. In *Evolutionary Computation (CEC), 2014 IEEE Congress on*, pages 2585–2592, July 2014.

[111] M. J. D. Powell. Algorithms for Nonlinear Constraints that use Lagrangian Functions. *Mathematical Programming*, 14:224–248, 1978.

[112] Inc. The MathWorks. Global optimization toolbox user's guide. Technical report, The MathWorks, Inc., 2013.

[113] Laurence D. Hoffmann, Gerald L. Bradley, and Kenneth H. Rosen. *Cálculo Aplicado*. McGrawHill, 2006.

[114] Huang V.L., A.K. Qin, and Suganthan P.N. Self-adaptive differential evolution algorithm for constrained real-parameter optimization. In *Evolutionary Computation, 2006. CEC 2006. IEEE Congress on*, pages 17–24, Vancouver, BC, Canada, July 2006. IEEE.

[115] Tasgetiren M.F. and Suganthan P.N. A multi-populated differential evolution algorithm for solving constrained optimization problem. In *Evolutionary Computation, 2006. CEC 2006. IEEE Congress on*, pages 33–40, Vancouver, BC, Canada, July 2006. IEEE.

[116] Kukkonen S. and Lampinen J. Constrained real-parameter optimization with generalized differential evolution. In *Evolutionary Computation, 2006. CEC 2006. IEEE Congress on*, pages 207–214, Vancouver, BC, Canada, July 2006. IEEE.

[117] Brest J., V. Zumer, and Maucec M.S. Self-adaptive differential evolution algorithm in constrained real-parameter optimization. In *Evolutionary Computation, 2006. CEC 2006. IEEE Congress on*, pages 215–222, Vancouver, BC, Canada, July 2006. IEEE.

[118] E. Mezura-Montes, J. Velazquez-Reyes, and C.A. Coello Coello. Modified differential evolution for constrained optimization. In *Evolutionary Computation, 2006. CEC 2006. IEEE Congress on*, pages 25–32, 2006.

[119] Saber M. Elsayed, Ruhul A. Sarker, and Daryl L. Essam. On an evolutionary approach for constrained optimization problem solving. *Applied Soft Computing*, 12(10):3208 – 3227, 2012.

[120] B. Tessema and G.G. Yen. An adaptive penalty formulation for constrained evolutionary optimization. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, 39(3):565–578, May 2009.

[121] Yong Wang, Zixing Cai, Yuren Zhou, and Wei Zeng. An adaptive tradeoff model for constrained evolutionary optimization. *Evolutionary Computation, IEEE Transactions on*, 12(1):80–92, Feb 2008.

[122] E. Mezura-Montes and Carlos A. Coello Coello. A simple multimembered evolution strategy to solve constrained optimization problems. *Evolutionary Computation, IEEE Transactions on*, 9(1):1–17, Feb 2005.

[123] R. L. Norton. *Diseño de Maquinaria, una Introducción a la Síntesis y al Análisis de Mecanismos y Máquinas*. McGrawHill, 1995.

[124] J.A. Cabrera, A. Simon, and M. Prado. Optimal synthesis of mechanisms with genetic algorithms. *Mechanism and Machine Theory*, 37(10):1165 – 1177, 2002.

[125] Radovan R. Bulatović and Stevan R. Dordević. On the optimum synthesis of a four-bar linkage using differential evolution and method of variable controlled deviations. *Mechanism and Machine Theory*, pages 235–246, 2009.

[126] Héctor Martínez-Alfaro. *Four-bar Mechanism Synthesis for n Desired Path Points Using Simulated Annealing*, chapter Advances in Metaheuristics for Hard Optimization. P. Siarry. Springer, Berlin, 2008.

[127] S.K. Acharyya and M. Mandal. Performance of {EAs} for four-bar linkage synthesis. *Mechanism and Machine Theory*, 44(9):1784 – 1794, 2009.

[128] H. Emdadi, M. Yazdanian, M.M. Ettefagh, and M.R. Feizi-Derakhshi. Double four-bar crank-slider mechanism dynamic balancing by meta-heuristic algorithms. *IJAIA*, (44):1–18, 2013.

[129] Álvaro Sánchez-Márquez, Eduardo Vega-Alvarado, Edgar Alfredo Portilla-Flores, and Efrén Mezura-Montes. Synthesis of a planar four-bar mechanism for position control using the harmony search algorithm. In *11TH International Conference on Electrical Engineering, Computing Science and Automatic Control (CCE)*, pages 1–6. IEEE, 2014.

[130] Manuel López-Ibáñez, Jéremie Dubois-Lacoste, Thomas Stützle, and Mauro Birattari. The irace package, iterated race for automatic algorithm configuration. Technical Report TR/IRIDIA/2011-004, IRIDIA, Université Libre de Bruxelles, Belgium, 2011.

[131] Betania Hernández-Ocaña, Ma. Del Pilar Pozos-Parra, and Efrén Mezura-Montes. Stepsize control on the modified bacterial foraging algorithm for constrained numerical optimization. In *Proceedings of the 2014 Conference on Genetic and Evolutionary Computation*, GECCO '14, pages 25–32, New York, NY, USA, 2014. ACM.

[132] Edgar Alfredo Portilla-Flores, Efrén Mezura-Montes, Jaime Alvarez-Gallegos, Carlos Artemio Coello-Coello, Carlos Alberto Cruz-Villar, and Miguel Gabriel Villarreal-Cervantes. Parametric reconfiguration improvement in non-iterative concurrent mechatronic design using an evolutionary-based approach. *Engineering Applications of Artificial Intelligence*, 24(5):757 – 771, 2011.

[133] Diet creator. Menu-creator. URL http://www.diet-creator.com/es, 2014. Accessed 11-06-2015.

[134] Nutrimind software. Nutrimind. URL http://www.nutrimind.net/, 2007. Accessed 11-06-2015.

[135] Lluvia Carolina Morales Reynaga. Pladiet: Un sistema de cómputo para el diseño de dietas individualizadas utilizando algoritmos genéticos. In *6TA Conferencia Iberoamericana en Sistemas Cibernéticos e Informática (CISCI 2007)*, pages 70 –75, 2007.

[136] R. Mallipeddi and P.N. Suganthan. Problem definitions and evaluation criteria for the cec 2010 competition on constrained real-parameter optimization. Technical report, School of EEE Nanyang Technological University, Singapore, April 2010.

[137] R. Pérez. *Análisis de Mecanismos y Problemas Resueltos.* Alfaomega Grupo Editor S.A. de C.V., 2006.

[138] J.E. Shigley and J.J. Uicker. *Teoría de Máquinas y Mecanismos.* McGraw-Hill, México, 1998.

[139] J. Mataix. *Tratado de Nutrición y Alimentación: Situaciones fisiológicas y patológicas.*, volume 2. Océano-Ergon, 2005.

[140] Rachel Berman. *Boosting your metabolism for dummies.* John Wiley & sons, Inc., 2013.

[141] Miriam Muñoz de Chávez, José Ángel Ledesma Solano, Adolfo Chávez Villasana, Fernando Pérez-Gil Romo, Eduardo Mendoza Martínez, and Concepción Calvo Carrillo. *Composición alimentos "Miriam Muñoz de Chávez"Valor nutritivo de los alimentos de mayor consumo.* McGraw-Hill Educación, México, 2 edition, 2010.

[142] Secretaría de Economía. Norma oficial mexicana para la promoción y educación para la salud en materia alimentaria: Nom-043.ssa2-2005. Technical report, Secretaría de Economía, 2005.

[143] Efrén Mezura-Montes and Carlos A. Coello Coello. Identifying on-line behavior and sources of difficulty in constrained optimization using evolutionary algorithms. In *In Proceedings of the IEEE Congress on Evolutionary Computation 2005 (CEC'2005)*, pages 1477–1484, Edinburgh, UK., 2005. IEEE Press.

[144] G.W. Corde and D.I. Foreman. *Nonparametric Statistics for Non-Statisticians: A Step-by-Step Approach.* John Wiley, Hoboken, NJ, 2009.

Anexos

Problemas de prueba

Dos conjuntos de **problemas de pruebas** son utilizadas para probar cada una de las propuestas derivadas de los experimentos realizados, las cuales son conocidas en el estado del arte. En la Tabla 8.1 se presenta un resumen de las características del primer conjunto de 24 funciones de prueba usadas en la competencia del CEC2006, su descripción completa puede encontrarse en [106]. El número máximo de evaluaciones (MAX_FEs) para resolver cada función de prueba es de 240,000 y se tiene una tolerancia de 0.0001 en la suma de violación en restricciones de igualdad. El mejor valor conocido $\boldsymbol{f(\overrightarrow{x^*})}$ para cada una de las funciones es presentado en la última columna de esta tabla. Además, n es el número de las variables, $\boldsymbol{\rho}$ es la relación estimada entre la región factible y el espacio de búsqueda, li es el número de restricciones de desigualdad lineales, ni es el número de restricciones de desigualdad no lineales, le es el número de restricciones de igualdad lineales, ne es el número de restricciones de igualdad no lineales, a es el número de restricciones activas y $\boldsymbol{f(\vec{x}^*)}$ es la mejor solución factible conocida.

Función	n	Tipo de función	ρ	LI	NI	LE	NE	a	$f(\vec{x}^*)$
g01	13	cuadrática	0.0003 %	9	0	0	0	6	-15
g02	20	no lineal	99.9973 %	2	0	0	0	1	-0.803619104
g03	10	no lineal	0.0026 %	0	0	0	1	1	-1.0005001
g04	5	cuadrática	27.0079 %	4	2	0	0	2	-30665.53867
g05	4	no lineal	0.0000 %	2	0	0	3	3	5126.496714
g06	2	no lineal	0.0057 %	0	2	0	0	2	-6961.813876
g07	10	cuadrática	0.0000 %	3	5	0	0	6	24.30620907
g08	2	no lineal	0.8581 %	0	2	0	0	0	-0.095825041
g09	7	no lineal	0.5199 %	0	4	0	0	2	680.6300574
g10	8	lineal	0.0020 %	6	0	0	0	6	7049.248021
g11	2	cuadrática	0.0973 %	0	0	0	1	1	0.7499
g12	3	cuadrática	4.7697 %	0	1	0	0	0	-1
g13	5	no lineal	0.0000 %	0	0	1	2	3	0.053941514
g14	10	no lineal	0.0000 %	0	0	3	0	3	-47.76488846
g15	3	cuadrática	0.0000 %	0	0	1	1	2	961.7150223
g16	5	no lineal	0.0204 %	4	34	0	0	4	-1.905155259
g17	6	no lineal	0.0000 %	0	0	0	4	4	8853.539675
g18	9	cuadrática	0.0000 %	0	13	0	0	6	-0.866025404
g19	15	no lineal	33.4761 %	0	5	0	0	0	32.65559295
g20	24	lineal	0.0000 %	0	6	2	12	16	0.2049794
g21	7	lineal	0.0000 %	0	1	0	5	6	193.7245101
g22	22	lineal	0.0000 %	0	1	8	11	19	236.4309755
g23	9	lineal	0.0000 %	0	2	3	1	6	-400.0551
g24	2	lineal	79.6556 %	0	2	0	0	2	-5.508013272

Tabla 8.1: Resumen de las características de las funciones de prueba del CEC2006.

El segundo conjunto de funciones de pruebas consta de 18 funciones, las cuales pueden ser ejecutadas con 10 y 30 dimensiones. La descripción completa de cada una de las funciones puede encontrarse en

[136]. El número máximo de evaluaciones para resolver cada función de prueba con 10 dimensiones es de 200,000 y con 30 dimensiones 600,000. Al igual que las funciones de prueba del CEC2006, se tiene una tolerancia de 0.0001 de violación en restricciones de igualdad. En la tabla 8.2 un resumen de las características de las funciones de prueba de este benchmark es presentado. Debido a que no se conoce la mejor solución a cada una de las funciones de prueba, se tomará como $\boldsymbol{f(\overrightarrow{x^*})}$ el mejor valor de todos los obtenidos por cada uno de los algoritmos presentes en una tabla de resultados. En la Tabla 8.2 se presenta la siguiente información: Región factible es la relación estimada entre la región factible y el espacio de búsqueda, I es el número de restricciones de desigualdad, E es el número de restricciones de igualdad y D es el número de variables de decisión.

Función	Rango de búsqueda	Tipo de objetivo	Número de restricciones		Región factible	
			E	I	$10n$	$30n$
C01	$[0,10]^D$	No separable	0	2 No separable	0.997689	1.000000
C02	$[-5.12, 5,12]^D$	Separable	1 Separable	2 Separable	0.000000	0.000000
C03	$[-1000,1000]^D$	No separable	1 Separable	0	0.000000	0.000000
C04	$[-50,50]^D$	Separable	2 No separable	0	0.000000	0.000000
C05	$[-600,600]^D$	Separable	2 Separable	0	0.000000	0.000000
C06	$[-600,600]^D$	Separable	2 Rotada	0	0.000000	0.000000
C07	$[-140,140]^D$	No separable	0	1 Separable	0.000000	0.000000
C08	$[-140,140]^D$	No separable	0	1 Rotada	0.505123	0.503725
C09	$[-500,500]^D$	No separable	1 Separable	0	0.379512	0.375278
C10	$[-500,500]^D$	No separable	1 Rotada	0	0.000000	0.000000
C11	$[-100,100]^D$	Rotada	1 No separable	0	0.000000	0.000000
C12	$[-1000,1000]^D$	Separable	1 No separable	1 Separable	0.000000	0.000000
C13	$[-500,500]^D$	Separable	0	2 Separable, 1 No separable	0.000000	0.000000
C14	$[-1000,1000]^D$	No separable	0	3 Separable	0.003112	0.006123
C15	$[-1000,1000]^D$	No separable	0	3 Rotada	0.003210	0.006023
C16	$[-10,10]^D$	No separable	2 Separable	1 Separable, 1 No separable	0.000000	0.000000
C17	$[-10,10]]^D$	No separable	1 Separable	2 No separable	0.000000	0.000000
C18	$[-50,50]^D$	No separable	1 Separable	1 Separable	0.000010	0.000000

Tabla 8.2: Características de las 18 funciones de prueba del CEC2010.

Problemas del mundo real

Problemas del mundo real también son considerados para probar el rendimiento de las propuestas derivadas de este trabajo de investigación. El primero corresponde a la síntesis de un mecanismo de cuatro barras del cual se obtienen tres casos diferentes y el segundo problema real la minimización del error entre las calorías requeridas por una persona y las calorías del menú nutricional encontrado por el algoritmo EMBFOA con un margen de error del 10 %.

Análisis del mecanismo de cuatro barras

En la Figura 8.1 se muestra un mecanismo planar de cuatro barras, con los elementos siguientes: barra de referencia ($\boldsymbol{r_1}$), manivela o barra de entrada ($\boldsymbol{r_2}$), acoplador o biela ($\boldsymbol{r_3}$), y balancín o barra de salida ($\boldsymbol{r_4}$). Para analizar este mecanismo se proponen dos sistemas coordenados: el sistema A, fijo al mundo real (0XY) y el sistema B de referencia ($\boldsymbol{OX_rY_r}$), donde $\boldsymbol{(x_0, y_0)}$ es la distancia entre los orígenes de ambos sistemas, $\boldsymbol{\theta_0}$ es el ángulo de rotación del sistema de referencia y $\boldsymbol{\theta_i}$ ($\boldsymbol{i = 2, 3, 4}$) corresponde a los ángulos de las barras del mecanismo; por último, las coordenadas $\boldsymbol{(r_{cx}, r_{cy})}$ determinan el punto $\boldsymbol{C}$ del acoplador.

El problema abordado en este trabajo es la síntesis de la longitud de un mecanismo de cuatro barras en tres casos de estudios diferentes de trayectoria y control de movimiento; en todas ellas el punto $\boldsymbol{C}$ del acoplador debe *pasar* a través de una serie de puntos predefinidos consecutivamente. La cinemática de los mecanismos de cuatro barras han sido tratados ampliamente, una explicación

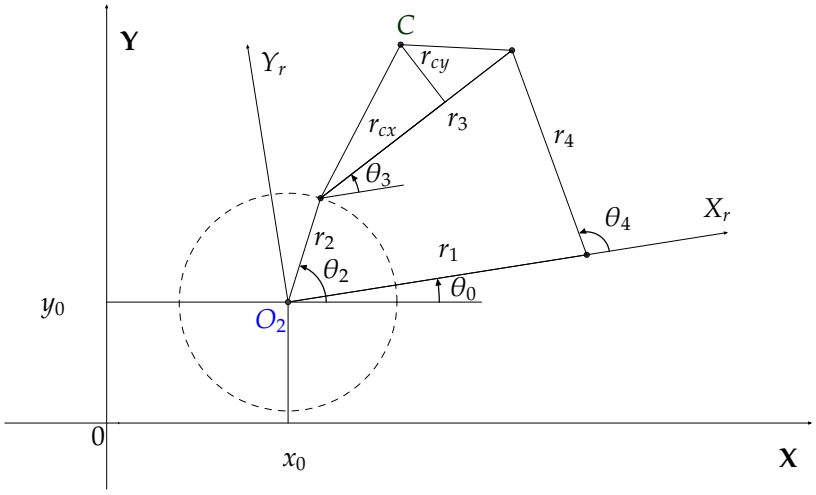

Figura 8.1: Mecanismo de cuatro barras

detallada se encuentra en [123, 137]; en este trabajo se considera el análisis de posición del mecanismo, así, se puede establecer la ecuación de cierre de circuito como:

$$\vec{r}_1 + \vec{r}_4 = \vec{r}_2 + \vec{r}_3 \tag{8.1}$$

Aplicando notación polar para cada término de (8.1), se obtiene

$$r_1 e^{j\theta_1} + r_4 e^{j\theta_4} = r_2 e^{j\theta_2} + r_3 e^{j\theta_3} \tag{8.2}$$

Utilizando la ecuación de Euler en (8.2) y agrupando términos reales e imaginarias:

$$\begin{aligned} r_1 cos\theta_1 + r_4 cos\theta_4 &= r_2 cos\theta_2 + r_3 cos\theta_3 \\ r_1 sin\theta_1 + r_4 sin\theta_4 &= r_2 sin\theta_2 + r_3 sin\theta_3 \end{aligned} \tag{8.3}$$

La posición angular $\boldsymbol{\theta_3}$ se puede obtener mediante la expresión de la parte izquierda del sistema de ecuaciones (8.3) en términos de $\boldsymbol{\theta_4}$:

$$\begin{aligned} r_4 cos\theta_4 &= r_2 cos\theta_2 + r_3 cos\theta_3 - r_1 cos\theta_1 \\ r_4 sin\theta_4 &= r_2 sin\theta_2 + r_3 sin\theta_3 - r_1 sin\theta_1 \end{aligned} \tag{8.4}$$

Por la cuadratura del sistema de ecuaciones (8.4) y la adición de sus términos, la ecuación de Freudenstein compactada se obtiene como:

$$A_1 cos\theta_3 + B_1 sin\theta_3 + C_1 = 0 \tag{8.5}$$

con:

$$A_1 = 2r_3 (r_2 cos\theta_2 - r_1 cos\theta_1) \tag{8.6}$$

$$B_1 = 2r_3 (r_2 sin\theta_2 - r_1 sin\theta_1) \tag{8.7}$$

$$C_1 = r_1^2 + r_2^2 + r_3^2 - r_4^2 - 2r_1 r_2 cos(\theta_1 - \theta_2) \tag{8.8}$$

El ángulo $\boldsymbol{\theta_3}$ se puede calcular en función de $\boldsymbol{A_1}$, $\boldsymbol{B_1}$, $\boldsymbol{C_1}$ y $\boldsymbol{\theta_2}$, si $\boldsymbol{sin\theta_3}$ y $\boldsymbol{cos\theta_3}$ son expresados en términos de $\boldsymbol{tan\,(\theta_3/2)}$:

$$\boldsymbol{sin\theta_3 = \frac{2tan(\theta_3/2)}{1+tan^2(\theta_3/2)}} \quad , \quad \boldsymbol{cos\theta_3 = \frac{1-tan^2(\theta_3/2)}{1+tan^2(\theta_3/2)}} \tag{8.9}$$

Una ecuación lineal de segundo orden se obtiene por sustitución en (8.5):

$$\boldsymbol{[C_1 - A_1]\,tan^2\left(\frac{\theta_3}{2}\right) + [2B_1]\,tan\left(\frac{\theta_3}{2}\right) + A_1 + C_1 = 0} \tag{8.10}$$

Derivado de (8.10), la posición angular $\boldsymbol{\theta_3}$ es dada por (8.11),

$$\boldsymbol{\theta_3 = 2arctan\left[\frac{-B_1 \pm \sqrt{B_1^2 + A_1^2 - C_1^2}}{C_1 - A_1}\right]} \tag{8.11}$$

Un proceso similar es llevado acabo para encontrar $\boldsymbol{\theta_4}$ [137]. El signo para el radical debe ser seleccionado en las equaciaciones $\boldsymbol{\theta_3}$ y $\boldsymbol{\theta_4}$ de acuerdo a la configuración del mecanismo de cuatro barras, como se muestra en la Tabla 8.3.

Configuración	$\boldsymbol{\theta_3}$	$\boldsymbol{\theta_4}$
open	$+\sqrt{}$	$-\sqrt{}$
crossed	$-\sqrt{}$	$+\sqrt{}$

Table 8.3: Signo del radical y tipo de mecanismo

El punto de interés del acoplador es $\boldsymbol{C}$, para determinar su posición en el sistema B ($\boldsymbol{OX_rY_r}$) que tiene que establecer que:

$$\begin{aligned} \boldsymbol{C_{xr}} &= \boldsymbol{r_2 cos\theta_2 + r_{cx} cos\theta_3 - r_{cy} sin\theta_3} \\ \boldsymbol{C_{yr}} &= \boldsymbol{r_2 sin\theta_2 + r_{cx} sin\theta_3 + r_{cy} cos\theta_3} \end{aligned} \tag{8.12}$$

En el sistema global de coordenadas, este punto es expresado como:

$$\begin{bmatrix} \boldsymbol{C_x} \\ \boldsymbol{C_y} \end{bmatrix} = \begin{bmatrix} \boldsymbol{cos\theta_0} & \boldsymbol{-sin\theta_0} \\ \boldsymbol{sin\theta_0} & \boldsymbol{cos\theta_0} \end{bmatrix} \begin{bmatrix} \boldsymbol{C_{xr}} \\ \boldsymbol{C_{yr}} \end{bmatrix} + \begin{bmatrix} \boldsymbol{x_0} \\ \boldsymbol{y_0} \end{bmatrix} \tag{8.13}$$

Como puede observarse, para las ecuaciones (8.12), (8.13) y las expresiones correspondientes al mecanismo cinemático es necesario calcular la posición del punto $\boldsymbol{C}$ a lo largo de una trayectoria.

Estrategías de optimización

Una vez que los mecanismos cinemáticos son establecidos el siguiente paso es definir el problema como un caso de optimización numérica, por lo que es necesario especificar las operaciones matemáticas y sus parámetros para la evaluación del sistema.

Diseño de restricciones

Cuando se trabaja con mecanismos de cuatro barras dos aspectos importantes relacionados a las restricciones sobre su rendimiento deben ser tomados en cuenta; esto corresponde al criterio de movilidad y dimensión:

Ley de Grashof: Define los criterios para asegurar movilidad completa al menos en una de las cuatro barras del mecanismo. Esta ley afirma que *para un eslabonamiento plano de cuatro barras, la suma de las longitudes más corta y más larga de los eslabones no debe ser mayor que la suma de las longitudes de los dos eslabones restantes, si se desea una rotación relativa continua entre dos elementos [138].* Si se denota como $\boldsymbol{s}$ a la longitud del eslabón más corto, $\boldsymbol{l}$ representa la barra más larga y $\boldsymbol{p}, \boldsymbol{q}$ para los eslabones restantes, se establece que:

$$\boldsymbol{l + s \leq p + q} \tag{8.14}$$

En este trabajo, la ley de Grashof está dada por:

$$\boldsymbol{r_1 + r_2 \leq r_3 + r_4} \tag{8.15}$$

Así, para asegurar que el método de solución cumple con esta ley, se establecen las siguientes restricciones:

$$\boldsymbol{r_2 < r_3, \quad r_3 < r_4, \quad r_4 < r_1} \tag{8.16}$$

Secuencia de entrada de ángulos: Cuando el problema de síntesis es la generación de la trayectoria o el movimiento sin sincronización prescrita, los valores de los ángulos de la manivela tienen que ser ordenados. Si el valor para el punto de precisión $\boldsymbol{i}$ se denota como $\boldsymbol{\theta_2^i}$, se requiere que:

$$\boldsymbol{\theta_2^1 < \theta_2^2 < ... < \theta_2^N} \tag{8.17}$$

donde $\boldsymbol{N}$ es el número de puntos calculados individualmente.

Función objetivo

La síntesis del mecanismo se lleva a cabo con el fin de calcular la longitud de sus barras, el ángulo de rotación con respecto al sistema de referencia, la distancia entre los sistemas de coordenadas, y el conjunto de ángulos para la barra de entrada para generar una trayectoria o un movimiento. En el sistema global de coordenadas ($\boldsymbol{OXY}$), el punto de precisión $\boldsymbol{i}$ se indica como:

$$\boldsymbol{C_d^i = \left[C_{xd}^i, C_{yd}^i\right]^T} \tag{8.18}$$

El conjunto de $\boldsymbol{N}$ puntos de precisión es definido como:

$$\boldsymbol{\Omega = \left\{C_d^i | i \in N\right\}} \tag{8.19}$$

Entonces, dado un conjunto de valores de las barras del mecanismo y sus parámetros $\boldsymbol{x_0, y_0, \theta_0}$, cada punto del acoplador puede ser expresado como una función de la posición de la barra de entrada:

$$\boldsymbol{C^i = \left[C_x(\theta_2^i), C_y(\theta_2^i)\right]^T} \tag{8.20}$$

En consecuencia, se desea reducir al mínimo la distancia entre los puntos de precisión individuales $\boldsymbol{C_d^i}$ y los puntos calculados $\boldsymbol{C^i}$. Para cuantificar tal distancia de todos los puntos de precisión, la siguiente función es propuesta:

$$\boldsymbol{f(\theta_2^i) = \sum_{i=1}^{N} \left[(C_{xd}^i - C_x^i)^2 + (C_{yd}^i - C_y^i)^2\right]} \tag{8.21}$$

Implementación Númerica

Caso de estudio $M01$: Seguimiento de trayectoria lineal

En este específico problema, el acoplador debe pasar a través de seis puntos de precisión definidos por las cardinalidades presentadas en (8.22), alineados en una trayectoria lineal vertical:

$$\boldsymbol{\Omega = \{(20,20),(20,25),(20,30),(20,35),(20,40),(20,45)\}} \tag{8.22}$$

El vector de variables de diseño es:

$$\boldsymbol{p = \{r_1, r_2, r_3, r_4, r_{cx}, r_{cy}, \theta_0, x_0, y_0, \theta_2^1, \theta_2^2, \theta_2^3, \theta_2^4, \theta_2^5, \theta_2^6\}} \tag{8.23}$$

donde las primeras cuatro variables corresponden a la longitug de las barras del mecanismo presentado en la Figura 8.1, las siguientes dos variables son la posición del acoplador, $\boldsymbol{\theta_0}$ es el ángulo de orientación del sistema con respecto a la horizontal y $\boldsymbol{O_2 = (x_0, y_0)}$ es su posición de coordenadas, y las últimas seis variables son la secuencia de los valores del ángulo para la barra de entrada $\boldsymbol{r_2}$. Valores superiores e inferiores para cada variable de diseño son definidos como:

$$\begin{aligned} \boldsymbol{r_1, r_2, r_3, r_4} &\in \boldsymbol{[0, 60]} \\ \boldsymbol{r_{cx}, r_{cy}, x_0, y_0} &\in \boldsymbol{[-60, 60]} \\ \boldsymbol{\theta_0, \theta_2^1, \theta_2^2, \theta_2^3, \theta_2^4, \theta_2^5, \theta_2^6} &\in \boldsymbol{[0, 2\pi]} \end{aligned}$$

Consideremos el problema de optimización numérica mono-objetivo descrito por (8.24) hasta (8.33) para obtener la solución a la síntesis de un mecanismo de cuatro barras para un seguimiento de trayectoria lineal.

$$\boldsymbol{\text{mín } f(p) = \sum_{i=1}^{N} [(C_{xd}^i - C_x^i)^2 + (C_{yd}^i - C_y^i)^2] p \in \mathbb{R}^{15}} \tag{8.24}$$

Sujeto a:

$$\boldsymbol{g_1(p) = r_1 + r_2 - r_3 - r_4 \leq 0} \tag{8.25}$$
$$\boldsymbol{g_2(p) = r_2 - r_3 \leq 0} \tag{8.26}$$
$$\boldsymbol{g_3(p) = r_3 - r_4 \leq 0} \tag{8.27}$$
$$\boldsymbol{g_4(p) = r_4 - r_1 \leq 0} \tag{8.28}$$
$$\boldsymbol{g_5(p) = \theta_2^1 - \theta_2^2 \leq 0} \tag{8.29}$$
$$\boldsymbol{g_6(p) = \theta_2^2 - \theta_2^3 \leq 0} \tag{8.30}$$
$$\boldsymbol{g_7(p) = \theta_2^3 - \theta_2^4 \leq 0} \tag{8.31}$$
$$\boldsymbol{g_8(p) = \theta_2^4 - \theta_2^5 \leq 0} \tag{8.32}$$
$$\boldsymbol{g_9(p) = \theta_2^5 - \theta_2^6 \leq 0} \tag{8.33}$$

Caso de estudio $M02$: Seguimiento de trayectoria con puntos no alineados

Para este caso, el seguimiento implica tiempo prescrito y la trayectoria es desaliniada, dada por el conjunto de puntos descritos en (8.34):

$$\boldsymbol{\Omega = \{(3,3),(2{,}759,3{,}363),(2{,}372,3{,}663),(1{,}890,3{,}862),(1{,}355,3{,}943)\}} \tag{8.34}$$

El vector de variables de diseño es:

$$p = \{r_1, r_2, r_3, r_4, r_{cx}, r_{cy}\} \tag{8.35}$$

Consideremos $x_0, y_0, \theta_0 = 0$. La restricción establecida por (8.17) no es aplicada para este caso, ya que existe una sincronización dada por la siguiente secuencia de ángulos de entrada:

$$\theta_2^i = \left\{\frac{\pi}{6}, \frac{\pi}{4}, \frac{\pi}{3}, \frac{10\pi}{24}, \frac{\pi}{2}\right\}$$

Los valores superiores e inferiores para el diseño de las variables son definidos como:

$$\begin{aligned} r1, r2, r3, r4 &\in [0, 50] \\ r_{cx}, r_{cy} &\in [-50, 50] \end{aligned} \tag{8.36}$$

El problema de optimización numérico mono-objetivo para este caso es descrito por (8.37) hasta (8.41)

$$\text{mín } f(p) = \sum_{i=1}^{N} [(C_{xd}^i - C_x^i)^2 + (C_{yd}^i - C_y^i)^2] p \in \mathbb{R}^6 \tag{8.37}$$

Sujeto a:

$$g_1(p) = r_1 + r_2 - r_3 - r_4 \leq 0 \tag{8.38}$$
$$g_2(p) = r_2 - r_3 \leq 0 \tag{8.39}$$
$$g_3(p) = r_3 - r_4 \leq 0 \tag{8.40}$$
$$g_4(p) = r_4 - r_1 \leq 0 \tag{8.41}$$

Caso de estudio $M03$: Generación de movimiento delimitado por un conjunto de pares de puntos

Este caso considera diez pares de punto de precisión dados por la cardinalidades presentadas en la Tabla 8.4.

Pares	C_{1d}	C_{2d}
1	(1.768, 2.3311)	(1.9592, 2.44973)
2	(1.947, 2.6271)	(2.168, 2.675)
3	(1.595, 2.7951)	(1.821, 2.804)
4	(1.019, 2.7241)	(1.244, 2.720)
5	(0.479, 2.4281)	(0.705, 2.437)
6	(0.126, 2.0521)	(0.346, 2.104)
7	(-0.001, 1.720)	(0.195, 1.833)
8	(0.103, 1.514)	(0.356, 1.680)
9	(0.442, 1.549)	(0.558, 1.742)
10	(1.055, 1.905)	(1.186, 2.088)

Table 8.4: Pares de puntos de precisión para el caso de estudio $M03$

El vector de diseño es:

$$p = \{r_1, r_2, r_3, r_4, r_{cx}, r_{cy}, \theta_0, x_0, y_0, \theta_2^1, ..., \theta_2^{10}\} \tag{8.42}$$

Con límites definidos como:

$$\begin{aligned} \boldsymbol{r1, r2, r3, r4} &\in \boldsymbol{[0, 60]} \\ \boldsymbol{r_{cx}, r_{cy}, x_0, y_0} &\in \boldsymbol{[-60, 60]} \\ \boldsymbol{\theta_0, \theta_2^1, ..., \theta_2^{10}} &\in \boldsymbol{[0, 2\pi]} \end{aligned}$$

Considerando que la trayectoria para este caso es definido por pares de puntos de precisión, la función objetivo en (8.21) tiene que ser modificada para considerar el error con respecto a ambos puntos en cada par. Así, la nueva función es dada por (8.43), y el correspondiente problema numérico mono-objetivo es descrito por (8.43) hasta (8.56):

$$\begin{aligned} &\textit{Min} \quad \boldsymbol{f(\vec{p}) = Error_1 + Error_2} \qquad (8.43) \\ &\boldsymbol{Error_1 = \sum_{i=1}^{K} \left[(C_{1xd}^i - C_x^i)^2 + (C_{1yd}^i - C_y^i)^2 \right]} \\ &\boldsymbol{Error_2 = \sum_{i=1}^{K} \left[(C_{2xd}^i - C_x^i)^2 + (C_{2yd}^i - C_y^i)^2 \right]} \end{aligned}$$

Sujeto a:

$$\begin{aligned} \boldsymbol{g_1(p)} &= \boldsymbol{r_1 + r_2 - r_3 - r_4 \leq 0} && (8.44) \\ \boldsymbol{g_2(p)} &= \boldsymbol{r_2 - r_3 \leq 0} && (8.45) \\ \boldsymbol{g_3(p)} &= \boldsymbol{r_3 - r_4 \leq 0} && (8.46) \\ \boldsymbol{g_4(p)} &= \boldsymbol{r_4 - r_1 \leq 0} && (8.47) \\ \boldsymbol{g_5(p)} &= \boldsymbol{\theta_2^1 - \theta_2^2 \leq 0} && (8.48) \\ \boldsymbol{g_6(p)} &= \boldsymbol{\theta_2^2 - \theta_2^3 \leq 0} && (8.49) \\ \boldsymbol{g_7(p)} &= \boldsymbol{\theta_2^3 - \theta_2^4 \leq 0} && (8.50) \\ \boldsymbol{g_8(p)} &= \boldsymbol{\theta_2^4 - \theta_2^5 \leq 0} && (8.51) \\ \boldsymbol{g_9(p)} &= \boldsymbol{\theta_2^5 - \theta_2^6 \leq 0} && (8.52) \\ \boldsymbol{g_{10}(p)} &= \boldsymbol{\theta_2^6 - \theta_2^7 \leq 0} && (8.53) \\ \boldsymbol{g_{11}(p)} &= \boldsymbol{\theta_2^7 - \theta_2^8 \leq 0} && (8.54) \\ \boldsymbol{g_{12}(p)} &= \boldsymbol{\theta_2^8 - \theta_2^9 \leq 0} && (8.55) \\ \boldsymbol{g_{13}(p)} &= \boldsymbol{\theta_2^9 - \theta_2^{10} \leq 0} && (8.56) \end{aligned}$$

Menú nutricional

Los menús nutricionales son planes de alimentación que constan de alimentos y sus diversas preparaciones. Cada elemento del menú es cuantificado en raciones, gramos o mililitros, lo que permite obtener una cuantificación global de las calorías y nutrientes del menú el cual debe satisfacer los requerimientos del usuario [139] para una alimentación correcta, es decir, que proporcione al organismo la energía y los nutrientes necesarios para realizar las actividades cotidianas y mantener las funciones vitales. Un menú también debe agregar el gasto termógeno de los alimentos y el gasto por actividad fisica que realiza un usuario.

En la literatura especializada se encuentran cuatro leyes de la alimentación que debe cumplir un menú:

- Ley de la Cantidad: se debe consumir la cantidad de energía que el cuerpo necesita.
- Ley de la Calidad: se deben consumir alimentos pertenecientes a todos los grupos de alimentos.
- Ley de la Armonía: guardar una relación entre los nutrientes ingeridos.
- Ley de la Adecuación: adecuar la alimentación a las necesidades nutritivas, sociales y psicológicas del usuario.

Para la generación del menú nutricional, en este trabajo se requieren especificaciones generales, una base de alimentos, un modelado matemático de la función objetivo y restricciones, un algoritmo adaptado para espacio de números enteros del EMBFOA y una interfaz gráfica.

Especificaciones generales para el modelado matemático

El menú nutricional considera las siguientes especificaciones, además de las leyes de la alimentación:

1. El menú nutricional será para personas entre 18 y 60 años de edad, sin sobre peso o desnutrición y sin patologías
2. El menú nutricional se compone de 23 alimentos agrupados en 5 comidas: (1) desayuno, (2) intermedio, (3) almuerzo, (4) intermedio y (5) cena
3. Los grupos de alimentos contemplados en este trabajo son 10: 1) verduras, 2) frutas, 3) cereales sin grasa, 4) cereales con grasa, 5) leguminosas, 6) alimentos de origen animal para el desayuno, 7) alimentos de origen animal general, 8) bebidas desayuno (puede contener leche), 9) bebidas generales y 10) acompañante (pan o tortilla)
4. La comida de intermedio es una colación entre el desayuno y el almuerzo y entre el almuerzo y la cena. El intermedio puede estar compuesto de un cereal con grasa y/o fruta y/o verdura
5. La comida desayuno puede estar compuesta por una bebida desayuno (puede contener leche) y/o cereales sin grasa y/o frutas y/o verduras y/o tortillas y/o alimentos de origen animal
6. La comida almuerzo puede estar compuesta por una bebida general (no contiene leche)y/o cereales sin grasa y/o frutas y/o verduras y/o tortillas y/o leguminosas y/o alimentos de origen animal
7. La comida cena puede estar compuesta por una bebida general (no contiene leche) y/o cereales sin grasa y/o frutas y/o verduras
8. El Requerimiento Energético diario de un Individuo (ReqEnergInd) se calculará usando la ecuación de Harris-Benedict revisada por Mifflin & St Jeor en [140] la cual se presenta en las ecuaciones 8.57 y 8.58 de pendiendo del sexo:

   Para el sexo femenino:

$$\boldsymbol{ReqEnergInd = 10peso + 6{,}25estatura - 5edad - 161} \tag{8.57}$$

   Para el sexo masculino:

$$\boldsymbol{ReqEnergInd = 10peso + 6{,}25estatura - 5edad + 5} \tag{8.58}$$

9. El efecto termógeno de los alimentos consiste en el gasto energético que se necesita para procesar los alimentos el cual corresponde al 10 % del ReqEnergInd y debe ser agregado a este mismo

10. El gasto por actividad física también se agrega al ReqEnergInd y los porcentajes varian dependiendo de la actividad:

Nivel de actividad	Ejemplo	% del ReqEnerInd
Reposo	Encamado	20 %
Ligero	Oficinista, profesional	37.5 %
Moderado	Ama de casa, estudiante	55 %
Intenso	Obrero, atleta	72.5&

Table 8.5: Porcentaje agregado al ReqEnergInd de acuerdo a la actividad física.

Base de alimentos para el menú nutricional

La base de alimentos fue construída en un archivo Excel donde los alimentos son agrupados en uno de los 10 grupos de alimentos de acuerdo a su origen. Para cada alimento se registra la información general y nutrimental, un ejemplo se muestra en la Tabla 8.6. Los alimentos almacenados provienen del libro [141].

Index	Nombre	Cantidad	Medida	Gramos/Mililitros	Ingredientes	Preparación	Calorías	Proteínas	Lípidos	Carbohidratos
6	Ensalada de jícama	1	taza	100g	Jícama	Cortar la jícama en tiras y servir	126	4.21	3.01	20.5

Table 8.6: Información general y nutrimiental registrada en la base de datos de alimentos en el grupo de frutas.

La base de alimentos cuenta con un número básico de alimentos para hacer las pruebas del algoritmo y generar los menús nutricionales. La Tabla 8.7 presenta el número de alimentos que contiene cada grupo en la base de alimentos, sin embargo, esta base puede ser incrementaba posteriormente por el usuario.

Grupo	Número de alimentos
Verduras	27
Frutas	48
Cereales sin grasa	55
Cereales con grasa	18
Leguminosas	27
Alimentos de origen animal para el desayuno	47
Alimentos de origen animal general	35
Bebidas para el desayuno	27
Bebidas generales	12
Acompañantes	4

Table 8.7: Número de alimentos por cada grupo en la base de alimentos.

Modelado matemático de la función objetivo y restricciones

La función objetivo (F. Obj) se presenta en la Ecuación 8.59 la cual busca minimizar el error entre las calorías requeridas por una persona y las calorías del menú nutricional encontrado por el algoritmo EMBFOA con un margen de error del 10 %:

$$\boldsymbol{E = |\sum_{m=1}^{c} KcalComida_m(\vec{k}) - ReqEnergInd|} \tag{8.59}$$

donde $\boldsymbol{c}$ es el número de comidas del menú (en este trabajo son 5), $\vec{\boldsymbol{k}}$ es un vector de números enteros donde cada valor hacen referencia a un index en la base de datos en un grupo de alimento específico, el $\vec{\boldsymbol{k}}$ tiene rangos distintos ya que depende del tipo de comida y al grupo de alimento. Basicamente, el algoritmo debe encontrar cinco vectores $\vec{\boldsymbol{k}}$ de números enteros y cada valor del vector debe estar entre el rango [1, $\boldsymbol{indexTgrupo_t}$], donde $\boldsymbol{indexTgrupo_t}$ es el número total de alimentos (indexs) en uno de los diez grupos de alimentos en la base de datos, $\boldsymbol{t = 1..,10}$.
$\boldsymbol{KcalComida_m(\vec{k})}$ son las calorias de cada alimento de las comidas del menú y $\boldsymbol{ReqEnergInd}$ es el requerimiento energético diario de un individuo el cual es calculado usando la ecuación 8.57 o 8.58 dependiendo del sexo y es agregado el efecto termógeno de los alimentos ($\boldsymbol{ReqEnergInd + 0{,}10ReqEnergInd}$) y el gasto por actividad física ($\boldsymbol{ReqEnergInd + \{0{,}20|0{,}375|0{,}55|0{,}725\} * ReqEnergInd}$).

Sujeta a las siguientes restricciones con un margen de error del 10 %:

La restricción $\boldsymbol{g_1}$ permite cumplir con la ley de la cantidad formulada en la ecuación 8.60.

$$\boldsymbol{g_1 = |\sum_{m=1}^{c} KcalComida_m(\vec{k}) - ReqEnergInd| - (1{,}10 ReqEnergInd - 0{,}90 ReqEnergInd) \leq 0} \tag{8.60}$$

Con las restricciones $\boldsymbol{g_2}$, $\boldsymbol{g_3}$ y $\boldsymbol{g_4}$ se permite cumplir con la ley de la armonía la cual establece que del aporte calórico diario, los carbohidratos deben cubrir entre el 55 y 60 %, las proteínas deben cubrir entre el 10 y 15 % y los lípidos entre el 25 y 30 %. La restricción $\boldsymbol{g_2}$ es calculada en la ecuación 8.61:

$$\boldsymbol{g_2 = \sum_{m=1}^{c} CarbComida_m(\vec{k}) \leq 1{,}10 CarbReq \quad \& \quad \sum_{m=1}^{c} CarbComida_m(\vec{k}) \geq 0{,}90 CarbReq} \tag{8.61}$$

donde $\boldsymbol{CarbComida_m(\vec{k})}$ son los carbohidratos totales de cada una de las comidas del menú y $\boldsymbol{CarbReq}$ son los carbohidratos requeridos por el individuo calculados con la ecuación 8.62.

$$\boldsymbol{CarbReq = (carb/100) \times (ReqEnergInd/4)} \tag{8.62}$$

donde $\boldsymbol{carb}$ es una variable aleatoria entre el rango [55, 60].

La restricción $\boldsymbol{g_3}$ es parte de la ley de la armonía la cual también estable que del aporte calórico diario, las proteínas deben cubrir entre el 10 y 15 %.

$$\boldsymbol{g_3 = \sum_{m=1}^{c} ProtComida_m(\vec{k}) \leq 1{,}10 ProtReq \quad \& \quad \sum_{m=1}^{c} ProtComida_m(\vec{k}) \geq 0{,}90 ProtReq} \tag{8.63}$$

donde $\boldsymbol{ProtComida_m(\vec{k})}$ son las proteínas totales de cada una de las comidas del menú y $\boldsymbol{ProtReq}$ son las proteínas requeridas por el individuo calculadas con la ecuación 8.64.

$$\boldsymbol{ProtReq = (prot/100) \times (ReqEnergInd/4)} \tag{8.64}$$

donde $\boldsymbol{prot}$ es una variable aleatoria entre el rango [10, 15].

La $\boldsymbol{g_4}$ es la última parte de la ley de la armonía la cual valida los lípidos del menú con la ecuación 8.65.

$$\boldsymbol{g_4 = \sum_{m=1}^{c} LipComida_m(\vec{k}) \leq 1{,}10 LipReq \quad \& \quad \sum_{m=1}^{c} LipComida_m(\vec{k}) \geq 0{,}90 LipReq} \tag{8.65}$$

donde $\boldsymbol{LipComida_m(\vec{k})}$ son los lípidos totales de cada una de las comidas del menú y $\boldsymbol{LipReq}$ son los lípidos requeridos por el individuo calculados con la ecuación 8.66.

$$\boldsymbol{LipReq = (lip/100) \times (ReqEnergInd/9)} \tag{8.66}$$

donde $\boldsymbol{lip}$ es una variable aleatoria entre el rango [25, 30].

La restricción $\boldsymbol{g_5}$ corresponde al requerimiento energético diario que debe consumir un individuo durante el desayuno. En este trabajo el porcentaje establecido para el desayuno es del 25 % y la restricción se calcula con la ecuación 8.67.

$$\boldsymbol{g_5 = \sum_{k=1}^{n} KcalComida_1(\vec{k}) \leq 1{,}10 Pdesay \quad \& \quad \sum_{k=1}^{n} KcalComida_1(\vec{k}) \geq 0{,}90 Pdesay} \tag{8.67}$$

donde $\boldsymbol{n}$ es el número de alimentos del desayuno, la especificación 5 para el modelado matemático menciona que son 6 alimentos contemplados en el desayuno de diferentes grupos de alimento. $\boldsymbol{KcalComida_1(\vec{k})}$ corresponde a las calorías de los alimentos en el desayuno (comida 1). $\boldsymbol{Pdesay}$ es el porcentaje de calorias que debe tener la comida desayuno, en este caso corresponde al 25 % del $\boldsymbol{ReqEnergInd}$.

La restricción $\boldsymbol{g_6}$ corresponde al requerimiento energético diario que debe consumir un individuo durante el amuerzo. En este trabajo el porcentaje establecido para el almuerzo es del 35 % y la restricción se calcula con la ecuación 8.68.

$$\boldsymbol{g_6 = \sum_{k=1}^{n} KcalComida_3(\vec{k}) \leq 1{,}10 Palmu \quad \& \quad \sum_{k=1}^{n} KcalComida_3(\vec{k}) \geq 0{,}90 Palmu} \tag{8.68}$$

donde $\boldsymbol{n}$ es el número de alimentos del almuerzo que en este trabajo son 7 los cuales fueron mencionados en la especificación 6. $\boldsymbol{KcalComida_3(\vec{k})}$ corresponde a las calorías de los alimentos en el almuerzo (comida 3). $\boldsymbol{Palmu}$ es el porcentaje de calorias que debe tener la comida almuerzo.

La restricción $\boldsymbol{g_7}$ corresponde al requerimiento energético diario que debe consumir un individuo durante la cena. En este trabajo el porcentaje establecido para la cena es del 20 % y la restricción se calcula con la ecuación 8.69.

$$\boldsymbol{g_7 = \sum_{k=1}^{n} KcalComida_5(\vec{k}) \leq 1{,}10 Pcena \quad \& \quad \sum_{k=1}^{n} KcalComida_5(\vec{k}) \geq 0{,}90 Pcena} \tag{8.69}$$

donde $\boldsymbol{n}$ son los alimentos del almuerzo que este trabajo son 4 los cuales fueron mencionados en la especificación 7. $\boldsymbol{KcalComida_5(\vec{k})}$ corresponde a las calorías de los alimentos en la cena (comida

5). $\boldsymbol{Pcena}$ es el porcentaje de calorias que debe tener la comida cena.

Por último, las retricciones $\boldsymbol{g_8}$ y $\boldsymbol{g_9}$ corresponden al requerimiento energético diario que debe consumir un individuo durante los intermedios. En este trabajo el porcentaje establecido para cada intermedio es el 10 %. Ambas restricciones se aplican de manera similar a cada uno de los intermedios. En la ecuación 8.70 se valida el intermedio consumido entre el desayuno y almuerzo y en la ecuación 8.71 se valida el intermedio consumido entre el almuerzo y cena.

$$\boldsymbol{g_8 = \sum_{k=1}^{n} KcalComida_2(\vec{k}) \leq 1{,}10 Pinter \quad \& \quad \sum_{k=1}^{n} KcalComida_2(\vec{k}) \geq 0{,}90 Pinter} \tag{8.70}$$

$$\boldsymbol{g_9 = \sum_{k=1}^{n} KcalComida_4(\vec{k}) \leq 1{,}10 Pinter \quad \& \quad \sum_{k=1}^{n} KcalComida_4(\vec{k}) \geq 0{,}90 Pinter} \tag{8.71}$$

donde $\boldsymbol{n}$ son los alimentos de cada uno de los intermedios, que este trabajo son 3 alimentos los cuales fueron mencionados en la especificación 4. $\boldsymbol{KcalComida_2(\vec{k})}$ y $\boldsymbol{KcalComida_4(\vec{k})}$ corresponde a las caloría de los alimentos en cada intermedio (comida 2 y 5). $\boldsymbol{Pinter}$ es el porcentaje de calorias que debe tener la comida intermedio.

En este modelado no se contemplan restricciones para la ley de la adecuación y la ley de la calidad, sin embargo para esta última ley se cuido que exitiera en el menú alimentos de los cinco grupos del plato del bien comer los cuales son: frutas, verduras, cerales, leguminosas y alimentos de origen animal [142].

Medidas de rendimiento

Las estadísticas básicas a utilizar son: mejor y peor resultado, y el valor de la media y desviación estándar (Desv.Est.) obtenido del conjunto de 25 ejecuciones independientes realizadas al algoritmo. Otras medidas a utilizar son las siguientes, donde las primeras cinco son tomadas de [106]:

- **Ejecución factible**: Una ejecución donde al menos una solución factible es encontrada dentro del MAX_FEs.
- **Ejecución exitosa**: Una ejecución donde al menos una solución factible $\vec{x}$ que satisface $\boldsymbol{f(\vec{x}) - f(\vec{x}^*) \leq 0{,}0001}$ es encontrada dentro del Max_FEs.
- **Tasa de factibilidad** = (número de ejecuciones factibles) / total de ejecuciones.
- **Tasa de éxito** = (número de ejecuciones exitosas) / total de ejecuciones.
- **Rendimiento exitoso**= media (evaluaciones por ejecuciones exitosas) × (# total de ejecuciones) / (# de ejecuciones exitosas).
- **Nado exitoso**: Un movimiento de nado donde la nueva posición es mejor (basado en las reglas de factibilidad) con respecto a la posición original de la bacteria.
- **Tasa de nados exitosos** = (# de nados exitosos) / total de nados, donde el total de nados=sb × nc × GMAX.

- **Medida de progreso (Progress ratio)**: Propuesto en [143], el objetivo es medir la capacidad de mejora del algoritmo dentro de la región factible del espacio de búsqueda. Para esta medida, se prefieren valores altos porque indican una mayor mejora de la primera solución factible encontrada. Se calcula como se muestra en la ecuación 8.72:

$$PR = \begin{cases} \left| In\sqrt{\frac{f_{min}(\theta^B(G_{ff}))}{f_{min}(\theta^B(GMAX))}} \right|, & \text{if } f_{min}(\theta^B(GMAX)) > 0 \\ \left| In\sqrt{\frac{f_{min}(\theta^B(G_{ff}))+1}{f_{min}(\theta^B(GMAX))+1}} \right|, & \text{if } f_{min}(\theta^B(GMAX)) = 0 \\ \left| In\sqrt{\frac{f_{min}(\theta^B(G_{ff}))+2|f_{min}(\theta^B(GMAX))|}{f_{min}(\theta^B(GMAX))+2|f_{min}(\theta^B(GMAX))|}} \right|, & \text{if } f_{min}(\theta^B(GMAX)) < 0 \end{cases} \tag{8.72}$$

  donde $f_{min}(\theta^B(G_{ff}))$ es el valor de la función objetivo de la primera solución factible encontrada por el algoritmo y $f_{min}(\theta^B(GMAX))$ es el valor de la función objetivo de la mejor solución encontrada al finalizar la ejecución del algoritmo. Estadísticas básicas como mejor, media, peor y desviación estándar pueden ser obtenidas.

- **Wilcoxon Signed Rank Test (WSRT)**: Prueba que permite juzgar la diferencia entre conjunto de resultados [144]. Los resultados se basan en los mejores valores de la aptitud de cada algoritmo en cada función de prueba. El signo '+'significa que el primer algoritmo es significativamente mejor que la segunda, '/'significa que el primer algoritmo es significativamente peor, y el signo '='cuando no hay ninguna diferencia significativa entre los dos algoritmos. El nivel de significación aplicada fue del 95 %. En todas las tablas, la propuesta derivada es considerada como el segundo algoritmo de comparación.

En general, el signo '-'se utiliza cuando no se proporciona un valor. Por otra parte, los valores en negrita representan el mejor valor en la tabla.

Las propuestas derivadas de MBFOA han sido codificadas usando Matlab R2009b y ejecutadas en una PC con un procesador Core 2 Duo 3.5, 4GB de RAM, y 64 bits del Sistema Operativo Windows 7.