Universidad de Granada
E.T.S.I Informática y Telecomunicaciones
Grado en Ingeniería Informática (GII)

Practica Alternativa al Examen: Mejora de una Metaheurística

Multi-Verse Optimizer: a nature-inspired algorithm for global optimization (2015)

Autores: Seyedali Mirjalili, Seyed Mihammad Mirjalili & Abdolreza Hatamlou

---

Metaheurísticas

Curso 2016-2017

Grupo 2: Viernes 17:30 - 19:30

Arthur Rodríguez Nesterenko - DNI Y1680851W

3 de julio de 2017

Índice

1. Introducción

Este trabajo alternativo al examen de teoría consistía en mejorar alguna/s de las metaheurísticas publicadas durante los ultimos años. Este conjunto de metaheurísticas propuestas por el profesor se basaban en distintos modelos evolutivos inspirados principalmente por **comportamientos observables en la naturaleza y el entorno en general**. El estudio y mejora que se ha realizado sobre la metaheurística elegida, **Multi-verse Optimizer**, comprende tres partes fundamentales que serán desarrolladas en las posteriores secciones.

Una primera parte constará del **análisis en bruto de la metaheurística**, la explicación del **algoritmo básico** y en base a qué fenónemo o comportamiento esta inspirado, realizando finalmente una **comparación** con distintos modelos de **Differential Evolution** proporcionados en distintos ficheros, todo esto en términos de un conjunto de funciones benchmark propuestas por el profesor.

La segunda parte de este estudio consiste en una **primera mejora** de la metaheurística a través del diseño de un **algoritmo memético**, utilizando una Local Search estándar y buscando el **mejor equilibrio posible entre los procesos de exploración y explotación**, midiendo además el número de evaluaciones que comprende cada uno de estos dos procesos. Los resultados pertinetes se obtienen con el mismo conjunto de funciones benchmark que en la primera parte.

Para la tercera y última parte del estudio se buscaba que el alumno propusiese distintas **mejoras potenciales de la metaheurística**, ya sea a través de parámetros adaptativos, componentes, balance optimo entre exploración y explotación, etc. Estas potenciales mejoras están destinadas a comprobar la **flexibilidad y adaptabilidad** del algoritmo en haras de obtener mejores resultados en cuanto a las funciones benchmark en caso de que éstas potenciales mejoras lo permitan. Sin más preámbulos comenzamos con el estudio de la metaheurística.

2. Análisis de la Metaheurística: Multi-verse Opimizer (MVO)

La primera sección de nuestra memoria constará de dos secciones claramente distinguidas: una primera sección en la que se realizará una **explicación breve y concisa del algoritmo** básico del autor [4] y otra en la que **compararemos los resultados** que ofrece la metaheurística en cuestión frente a diferentes técnicas de **Differential Evolution**, tanto las proporcionadas en el fichero *Tanabe* como las dos implementaciones de Daniel Molina de los dos algoritmos básicos de DE. El objetivo de este apartado consiste en **introducir una solución a un conjunto de problemas** (funciones benchmark) y comprobar su comportamiento frente a otros algoritmos con más rodaje en el camino.

2.1. Análisis de la MH: algoritmo básico de MVO

La idea del algoritmo Multi-verse Optimizer [4] surge a partir de una reciente teoría llamada **teoría de multiversos** en donde, contrario a la teoría de un único universo, existe o ha existido mas de un Big Bang a lo largo del tiempo y cada uno de estos Big Bang ha causado el **nacimiento de un universo distinto**, con muy probablemente, leyes de la física distintas para cada uno de estos. La teoría de los multiversos sostiene que este conjunto de universos **interactúan entre si y pueden incluso colisionar** entre ellos, sentando una sólida base para el desarrollo de nuestro algoritmo.

Para concretar un modelo matemático y físico para nuestro algoritmo, se han seleccionado 3 componentes indispensables en la teoría de los multiversos: *los agujeros blancos, los agujeros negros y los agujeros de gusano*. Un **agujero blanco**, a pesar de no haber sido visto aún en nuestro universo, es considerado por los físicos como un Big Bang (el origen de todo) y también se sostiene que estos elementos se crean donde ocurren **colisiones entre universos paralelos**. Un **agujero negro** por contrapartida **consume todo a su paso**, atrayendo cualquier tipo de masa o energía a través de su potente campo gravitacional. Finalmente los **agujeros de gusano** se consideran como túneles espacio-temporales a través de los cuales es posible que **un objeto pase de un universo a otro**.

Estos tres componentes principales (ver figura 2.1) permiten a los multiversos interactuar entre sí con el objetivo de llegar a una **situación estable**, principio que será la clave en la modelización matemática del algoritmo y su posterior puesta a prueba. Para comprender en mayor medida el comportamiento de nuestro algoritmo tenemos que pasar por distintas fases que nos permitirán conocer el trasfondo tanto físico como matemático que da paso a un algoritmo robusto y con grandes capacidades.

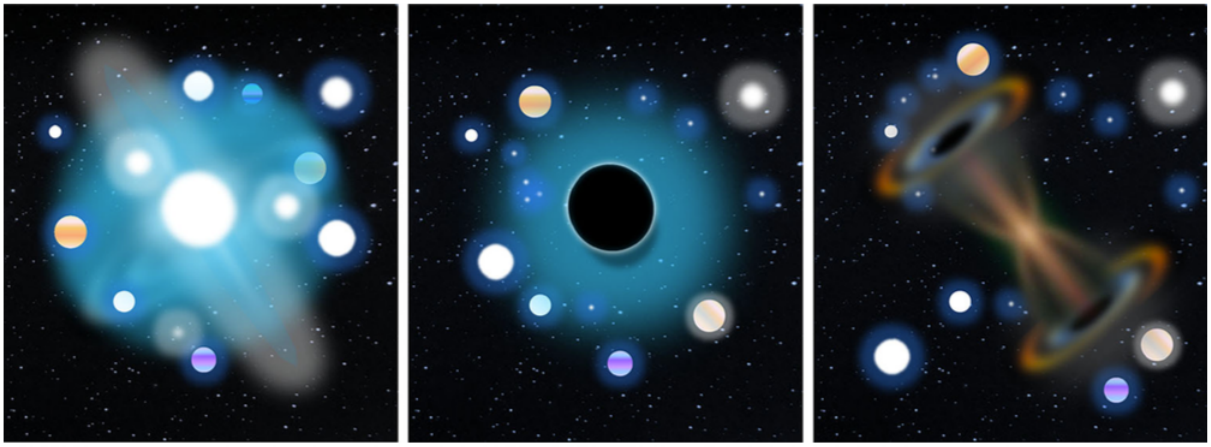

Figura 2.1: Agujero blanco, agujero negro y agujero de gusano, respectivamente

2.1.1. Definición del Algoritmo MVO: componentes, reglas y comportamiento

El algoritmo MVO propone un modelo evolutivo en donde los procesos de exploración y explotación sean llevados a cabo por los conceptos previamente descritos. Así, los *agujeros blancos y negros* serán los encargados de controlar la **exploración del espacio de soluciones** mientras que los *agujeros de gusano* realizarán el proceso de **explotación** de las soluciones más prometedoras.

Las componentes principales del algoritmo se describen a continuación, donde se representan las analogías del problema con la teoría de los multiversos:

1. **Soluciones**: en nuestro caso una solución será representada como un **universo** con objetos.
2. **Variables**: los objetos de los universos serán considerados como las variables de nuestro problema.
3. **Valor de la función objetivo**: será modelado como el **ritmo de inflación** del universo.
4. **Iteraciones del algoritmo**: hablaremos de **tiempo** medido en años luz.

Existen además un conjunto de reglas que son esenciales para describir el comportamiento del algoritmo en términos de todos y cada uno de los componentes utilizados, así como en los procesos de exploración y explotación asociados.

1. Cuanto **mayor** sea el **ritmo de inflación (IR)** de un universo, **mayor probabilidad** de que éste tenga un agujero blanco y **menor** de que tenga un agujero negro.
2. Los universos con mayor IR tienden a **enviar objetos** a través de los agujeros blancos a aquellos universos con menor IR. Estos objetos son **recibidos por los agujeros negros**.
3. Independientemente del IR de un universo, este puede experimentar un **movimiento de acercamiento al mejor universo** creado hasta la fecha a través de los agujeros de gusano.
4. Los distintos movimientos de cada universo con respecto a los demás están destinados a garantizar que **la media de los IR aumente** en el conjunto total de universos.

   El psedocódigo de la exploración se puede ver en la sección 3.2 del paper [4]. No se ha incluido en la memoria para evitar extender la misma de forma innecesaria.

El mecanismo de intercambio de objetos se puede ver en la figura 2.4, donde para el proceso de exploración se utiliza un mecanismo de ruleta para seleccionar el universo que tendrá un agujero blanco (en función del IR de cada universo) y en donde entran

en juego dos coeficientes principales que dan sentido al intercambio de objetos entre un universo y el mejor universo creado hasta el momento.

1. **WEP (Wormhole Existance Probability)**: modela la probabilidad de existencia de un agujero de gusano. Se incrementa de forma lineal para enfatizar en el proceso de explotación conforme el paso del tiempo.

$$\text{WEP} = \min + l \times \left(\frac{\max - \min}{L}\right)$$

Figura 2.2: WEP: min y max son los valores mínimos del paper (0.2 y 1 respectivamente), y l representa la iteración/tiempo actual y L el tiempo total-

2. **TDR (Travelling Distance Rate)**: medida de la distancia a la que un objeto puede ser teletransportado a través de un agujero de gusano. Se reduce en cada iteración.

$$\text{TDR} = 1 - \frac{l^{1/p}}{L^{1/p}}$$

Figura 2.3: TDR: p es un coeficiente que determina la capacidad de exploración.

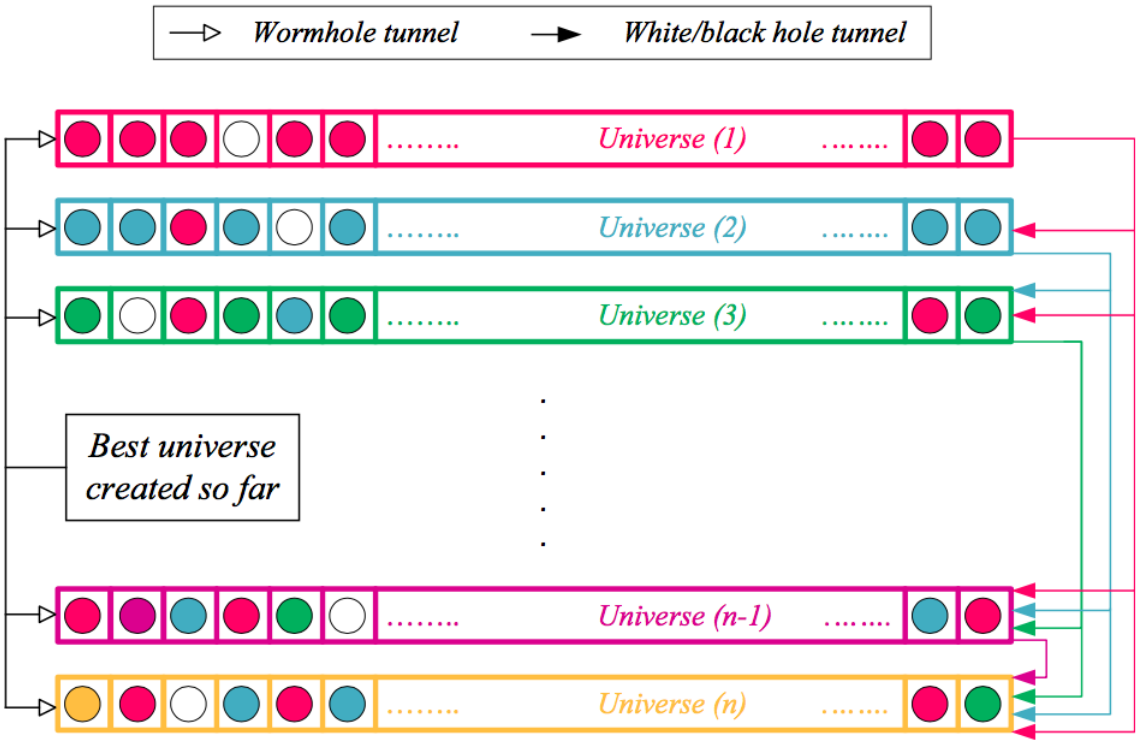

Figura 2.4: Mecanismo de intercambio de objetos

El pseudocódigo de la fase de explotación se encuentra en la sección 3.2 del paper y en la presentación en transparencias del estudio. Una vez comprendido como funciona el algoritmo básico vamos a pasar a realizar las comparaciones pertinentes frente a los dos algoritmos de DE implementados por Daniel Molina y a los resultados del fichero *Tanabe*, explicando los mismos y aportando conclusiones frente a lo que se ha obtenido.

2.1.2. Comparación del algoritmo MVO: DE y fichero Tanabe

La comparativa que se expone a continuación se basa en un conjunto de ejecuciones del algoritmo básico del autor, obtenido a través de su web personal [2] y en un conjunto de funciones benchmark correspondientes a las utilizadas en el CEC 2014 [3]. Los parámetros de las distintas ejecuciones son las mismas para todas las funciones: **dimensiones** del problema **10 y 30**, número de **evaluaciones** de la función objetivo igual a **10000*Dimension del problema** y **25 ejecuciones por cada función objetivo** para trabajar con valores medios. Se proponen una serie de tablas y gráficas comparativas para apreciar y comentar los resultados obtenidos.

1. Comparación de MVO con los resultados de los algoritmos de DE proporcionados por Daniel Molina. En color verde, aquellas funciones donde MVO obtiene el mínimo valor.

	DEBin	DEexp	MVO
F1	0,00E+00	0,00E+00	1,54E+05
F2	0,00E+00	0,00E+00	4,07E+03
F3	0,00E+00	0,00E+00	1,43E+01
F4	2,16E+01	2,19E+01	2,01E+01
F5	2,02E+01	2,01E+01	2,00E+01
F6	5,84E-01	1,84E-01	1,55E+00
F7	3,66E-02	3,58E-02	2,99E-01
F8	4,35E+00	3,98E-02	1,59E+01
F9	1,20E+01	8,79E+00	1,66E+01
F10	5,20E+01	1,15E+01	5,65E+02
F11	4,44E+02	5,55E+02	6,35E+02
F12	4,26E-01	4,39E-01	1,58E-01
F13	1,14E-01	1,12E-01	1,67E-01
F14	1,76E-01	1,63E-01	1,01E-01
F15	1,77E+00	1,24E+00	1,23E+00
F16	2,34E+00	2,35E+00	2,49E+00
F17	1,64E+01	8,49E+00	3,47E+03
F18	5,40E-01	5,54E-01	1,27E+04
F19	3,11E-01	4,53E-01	2,30E+00
F20	2,04E-01	6,60E-02	1,04E+02

(a) Comparación de resultados: 20 primeras funciones del CEC 2014, dimensión 10.

	DEBin	DEexp	MVO
F1	8,88E+04	2,95E+05	3,19E+06
F2	2,00E+02	2,00E+02	2,52E+04
F3	3,00E+02	3,00E+02	4,20E+02
F4	3,85E+02	4,11E+02	4,86E+02
F5	5,01E+02	5,00E+02	5,20E+02
F6	6,03E+02	6,17E+02	6,09E+02
F7	7,00E+02	7,00E+02	7,00E+02
F8	8,10E+02	8,00E+02	8,83E+02
F9	1,02E+03	9,73E+02	9,95E+02
F10	1,43E+03	9,93E+02	3,82E+03
F11	6,76E+03	4,31E+03	4,17E+03
F12	1,20E+03	1,20E+03	1,20E+03
F13	1,30E+03	1,30E+03	1,30E+03
F14	1,40E+03	1,40E+03	1,40E+03
F15	1,51E+03	1,51E+03	1,51E+03
F16	1,61E+03	1,61E+03	1,61E+03
F17	4,95E+03	3,93E+03	1,22E+05
F18	1,82E+03	1,84E+03	1,51E+04
F19	1,90E+03	1,91E+03	1,91E+03
F20	2,01E+03	2,03E+03	2,38E+03

(b) Comparación de resultados: 20 primeras funciones del CEC 2014, dimensión 30.

Para realizar una comparativa mas visual, se proponen una serie de gráficas para aclarar los resultados.

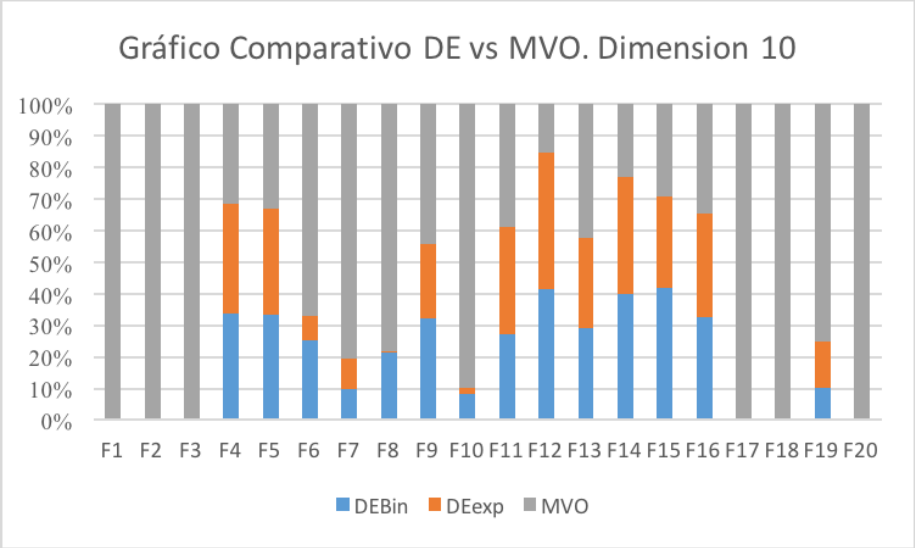

Figura 2.6: Gráfica MVO vs DE en dimensión 10

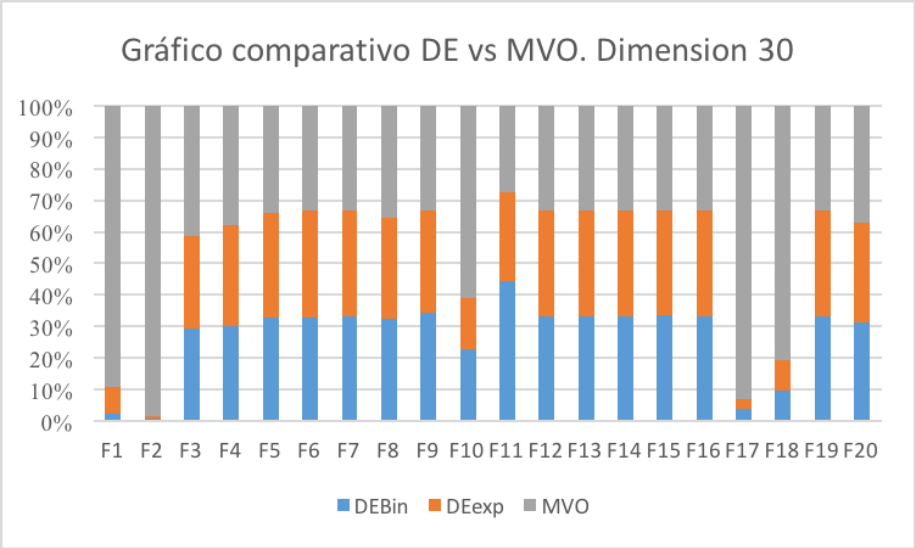

Figura 2.7: Gráfica MVO vs DE en dimensión 30

A la vista de los resultados se puede decir que MVO es un algoritmo que, por lo pronto, presenta un comportamiento ligeramente mas ajustado en **dimensiones mayores**, principalmente porque el número de iteraciones/evaluaciones de la función objetivo de las que dispone le permite acercarse más a un mínimo global. Si se comparan los resultados que no están destacados en verde se puede comprobar como no es capaz de hacer frente, en general, a un DE dado que los mecanismos que controlan la **exploración y explotación** no parecen ser tan potentes a priori como los que ofrece DE. Sobre estos aspectos incideremos más adelante en busca de potenciales mejoras.

2. **Comparación de MVO con los resultados de los algoritmos del fichero *Tanabe***: otra comparativa con más algoritmos de DE que superan enormemente la capacidad de nuestro algoritmo en la gran mayoría de funciones, ajustándose en algunos casos a los mejores valores conseguidos por estos algoritmos de DE.

	CoDE	D-SHADE	EPSDE	JADE	L-SHADE	POP-aCMA	SHADE11	SaDE	dynNP-jDE	iCMAES-ILS	MVO
F1	0,00E+00	0,00E+00	0,00E+00	0,00E+00	0,00E+00	0,00E+00	0,00E+00	2,55E+00	2,17E-07	0,00E+00	1,54E+05
F2	0,00E+00	0,00E+00	0,00E+00	0,00E+00	0,00E+00	0,00E+00	0,00E+00	0,00E+00	0,00E+00	0,00E+00	4,07E+03
F3	0,00E+00	0,00E+00	0,00E+00	6,17E-03	0,00E+00	0,00E+00	0,00E+00	0,00E+00	0,00E+00	0,00E+00	1,43E+01
F4	1,05E+01	3,08E+01	0,00E+00	2,76E+01	2,94E+01	2,82E+00	2,95E+01	1,81E+01	3,32E+00	1,44E+01	2,01E+01
F5	1,84E+01	1,77E+01	2,00E+01	1,73E+01	1,41E+01	1,81E+01	1,80E+01	1,58E+01	1,60E+01	1,47E+01	2,00E+01
F6	1,65E-06	0,00E+00	3,04E+00	1,76E-01	1,75E-02	3,31E-01	0,00E+00	0,00E+00	0,00E+00	0,00E+00	1,55E+00
F7	3,76E-02	5,31E-03	1,76E-02	1,19E-02	3,04E-03	0,00E+00	9,78E-03	7,24E-03	4,97E-03	0,00E+00	2,99E-01
F8	0,00E+00	0,00E+00	0,00E+00	0,00E+00	0,00E+00	3,70E+00	0,00E+00	0,00E+00	0,00E+00	2,54E-01	1,59E+01
F9	3,88E+00	3,08E+00	3,69E+00	3,51E+00	2,34E+00	3,26E-01	3,14E+00	3,58E+00	3,86E+00	9,75E-02	1,66E+01
F10	3,55E-02	4,90E-02	4,41E-02	6,12E-03	8,57E-03	9,16E+01	1,22E-02	1,96E-02	2,45E-03	1,22E+02	5,65E+02
F11	7,60E+01	5,49E+01	3,23E+02	8,37E+01	3,21E+01	1,17E+02	6,32E+01	1,96E+02	1,36E+02	8,59E+00	6,35E+02
F12	4,34E-02	5,29E-02	3,21E-01	2,50E-01	6,82E-02	1,01E-02	1,36E-01	4,35E-01	3,11E-01	6,50E-02	1,58E-01
F13	7,98E-02	4,89E-02	1,22E-01	8,40E-02	5,16E-02	1,09E-02	7,40E-02	1,25E-01	1,19E-01	9,11E-03	1,67E-01
F14	1,07E-01	9,01E-02	1,36E-01	1,11E-01	8,14E-02	2,82E-01	1,06E-01	1,86E-01	1,35E-01	1,55E-01	1,01E-01
F15	6,52E-01	4,03E-01	7,54E-01	5,78E-01	3,66E-01	5,47E-01	5,05E-01	7,90E-01	7,82E-01	7,23E-01	1,23E+00
F16	1,13E+00	1,34E+00	2,54E+00	1,65E+00	1,24E+00	2,53E+00	1,56E+00	1,97E+00	1,59E+00	1,91E+00	2,49E+00
F17	2,66E+00	3,38E+00	5,33E+01	3,09E+01	9,77E-01	3,89E+01	1,56E+00	2,83E+01	2,62E+00	2,10E+01	3,47E+03
F18	4,31E-01	4,75E-01	1,20E+00	2,39E-01	2,44E-01	3,58E+00	2,37E-01	1,65E+00	4,41E-01	5,26E-01	1,27E+04
F19	7,45E-02	2,05E-01	1,43E+00	2,55E-01	7,73E-02	8,28E-01	1,92E-01	6,69E-02	1,22E-01	7,08E-01	2,30E+00
F20	2,39E-02	2,73E-01	1,65E-01	3,24E-01	1,85E-01	1,32E+00	2,43E-01	1,08E-01	4,15E-02	8,04E-01	1,04E+02

Figura 2.8: Comparación de resultados fichero Tanabe (DE): 20 primeras funciones del CEC 2014, dimensión 10.

	CoDE	D-SHADE	EPSDE	JADE	L-SHADE	POP-aCMA	SHADE11	SaDE	dynNP-jDE	iCMAES-ILS	MVO
F1	2,63E+04	5,04E-03	2,42E+04	4,48E+02	0,00E+00	0,00E+00	4,81E+02	2,99E+05	4,65E+04	0,00E+00	3,19E+06
F2	0,00E+00	0,00E+00	0,00E+00	0,00E+00	0,00E+00	0,00E+00	0,00E+00	0,00E+00	0,00E+00	0,00E+00	2,52E+04
F3	0,00E+00	0,00E+00	0,00E+00	5,63E-04	0,00E+00	0,00E+00	0,00E+00	1,43E+01	0,00E+00	0,00E+00	4,20E+02
F4	2,52E+00	5,03E-09	3,21E+00	0,00E+00	0,00E+00	0,00E+00	0,00E+00	3,72E+01	2,04E+00	0,00E+00	4,86E+02
F5	2,01E+01	2,00E+01	2,03E+01	2,03E+01	2,01E+01	2,05E+01	2,01E+01	2,05E+01	2,03E+01	2,00E+01	5,20E+02
F6	1,99E+00	5,92E-02	1,89E+01	9,42E+00	1,38E-07	7,14E-01	5,29E-01	5,46E+00	1,20E+00	4,00E-03	6,09E+02
F7	1,45E-04	0,00E+00	2,08E-03	0,00E+00	0,00E+00	0,00E+00	4,83E-04	1,23E-02	0,00E+00	0,00E+00	7,00E+02
F8	0,00E+00	0,00E+00	0,00E+00	0,00E+00	0,00E+00	9,98E+00	0,00E+00	7,80E-02	0,00E+00	2,42E+00	8,83E+02
F9	4,04E+01	8,70E+00	4,44E+01	2,62E+01	6,78E+00	3,24E+00	1,58E+01	3,81E+01	3,39E+01	2,57E+00	9,95E+02
F10	5,00E-01	3,51E-02	2,01E-01	5,31E-03	1,63E-02	6,36E+02	1,27E-02	2,69E-01	4,08E-03	1,45E+02	3,82E+03
F11	1,95E+03	1,30E+03	3,56E+03	1,64E+03	1,23E+03	7,31E+02	1,40E+03	3,15E+03	1,95E+03	7,38E+01	4,17E+03
F12	6,00E-02	9,67E-02	5,25E-01	2,71E-01	1,61E-01	1,32E-02	1,62E-01	7,94E-01	3,62E-01	2,83E-02	1,20E+03
F13	2,31E-01	1,34E-01	2,43E-01	2,20E-01	1,24E-01	3,89E-02	2,04E-01	2,52E-01	2,53E-01	2,95E-02	1,30E+03
F14	2,39E-01	2,32E-01	2,78E-01	2,34E-01	2,42E-01	3,28E-01	2,25E-01	2,29E-01	2,66E-01	1,70E-01	1,40E+03
F15	3,18E+00	1,89E+00	5,67E+00	3,10E+00	2,15E+00	2,14E+00	2,56E+00	4,14E+00	4,76E+00	2,51E+00	1,51E+03
F16	9,26E+00	8,52E+00	1,11E+01	9,37E+00	8,50E+00	1,06E+01	9,15E+00	1,09E+01	9,22E+00	1,09E+01	1,61E+03
F17	1,45E+03	2,10E+02	4,61E+04	9,67E+03	1,88E+02	8,52E+02	1,06E+03	1,15E+04	9,58E+02	1,05E+03	1,22E+05
F18	1,34E+01	1,04E+01	3,32E+02	3,58E+02	5,91E+00	1,15E+02	4,99E+01	4,44E+02	2,10E+01	9,61E+01	1,51E+04
F19	2,70E+00	3,53E+00	1,33E+01	4,44E+00	3,68E+00	5,70E+00	4,31E+00	4,00E+00	3,91E+00	6,46E+00	1,91E+03
F20	1,09E+01	4,20E+00	5,00E+01	2,89E+03	3,08E+00	2,40E+01	1,26E+01	1,25E+02	8,53E+00	3,35E+01	2,38E+03

Figura 2.9: Comparación de resultados fichero Tanabe (DE): 20 primeras funciones del CEC 2014, dimensión 30.

3. Diseño de Algoritmo Memético: uso de Local Search

La segunda parte de nuestro estudio consiste en el diseño de un algoritmo memético utilizando la metaheurística elegida para el problema. El profesor nos proporcionaba un fichero zip con tres de los métodos de búsqueda locales estándar más utilizado actualmente: **Solis Wets, Simplex y CMAES.** El procedimiento que se ha seguido para desarrollar esta sección ha transcurrido por diversos caminos que comentaremos a continuación.

Tras obtener el código del algoritmo de la web del autor [2] y ejecutar la primera fase de nuestro estudio, pasamos a considerar cuál de los métodos debiamos utilizar para aplicarle una Local Search a nuestro algoritmo con el fin de optimizar los resultados obtenidos por el algoritmo básico. La **incompatibilidad** del código del autor con las LS proporcionadas por el profesor en cuanto a lenguaje de programación (el código del autor está escrito completamente en MATLAB mientra que las LS están en código C) fue el principal motivo por el cual decidimos ocupar la búsqueda de alguno de estos métodos ya implementados en lenguaje MATLAB.

En la página oficial de MathWorks, sección de *File Exchange* encontramos una implementación del algoritmo Covariance Matrix Adaptation Evolution Strategy (CMA-ES) [1] nativo, por lo que nuestra siguiente tarea pasaría por adaptar el algoritmo para que funcionase como una Local Search dentro del algoritmo MVO con el objetivo de optimizar las soluciones obtenidas por el algoritmo básico.

El algoritmo original, al ser un algoritmo evolutivo, consideraba una población inicial de soluciones a partir de la cual generaría mejores soluciones en cada iteración. Dado que nuestro objetivo era utilizar este potente algoritmo como una búsqueda local, optamos por realizar una **serie de modificaciones** en el mismo para garantizar que su desempeño fuese el máximo posible aún sin llegar a entrar en detalles de implementación a bajo nivel o a un conocimiento completo de cada uno de los cálculos del algoritmo. A continuación se detallan las modificaciones realizadas.

1. El algoritmo recibe una serie de parámetros de entrada: un **universo** (el **mejor** creado hasta la fecha), el **valor de la función** de evaluación para ese universo, la **función objetivo** (un índice en este caso), la **dimensión** del problema, las **cotas** superior e inferiores de las variables, el coeficiente **TDR** explicado anteriormente y el número **maximo de iteraciones** del algoritmo como LS.

2. Mientras que el resto de parámetros iniciales del algoritmo no se modifican a priori, el número de descendientes de la población se limita a un 50 % de la dimensión del problema (5 para dimension 10, 15 para dimensión 30). Con esto se busca mantener un adecuado equilibrio entre **exploración y explotación**.

3. Cuando el algoritmo empieza su ejecución, cada descendiente parte de la mejor so-

lución encontrada hasta el momento (universo inicial al principio) y se calcula como el valor de la característica de la mejor solución mas un factor calculado en base al TDR y al valor sigma (*step-size*) del universo en cuestión (fijado inicialmente a 0.3*(Ubound-LBound)). Ver código para aclarar cualquier duda.

4. Finalmente el algoritmo calcula la mejor solución, actualiza las medias, el denominado *step-size* y la matriz de covarianzas para devolver finalmente en una estructura el mejor universo creado y el valor de la función objetivo, junto con el número de llamadas a la función objetivo por ejecución de la LS.

Una vez realizadas las adaptaciones del algoritmo para que el mismo pudiese funcionar como una LS, se prodece a ejecutar una serie de pruebas preliminares para determinar la calidad de las soluciones que es capaz de proporcionar este algoritmo. Para ello sin embargo se requiere modificar ligeramente el algoritmo MVO para que acepte una búsqueda local que intente optimizar sus soluciones, y se dice *intente* puesto que no existe absoluta certeza de que el algoritmo y sus resultados se vean mejorados por esta implementación de la búsqueda local por distintas razones.

Para cada una de las pruebas realizadas se proponen una serie de restricciones y valores de los parámetros para establecer un punto de partida común a partir del cual podamos decidir si la LS es capaz de optimizar las soluciones de nuestro algoritmo. Si bien no se muestran todos los resultados empíricos de todas las pruebas, si se describen y se comentan los resultados obtenidos para no extender de sobremanera la memoria.

3.1. Prueba 1: Modificaciones del algoritmo básico

Para comenzar a utilizar CMAES como un método de LS hace falta definir una serie de parámetros que controlan el proceso de explotación. Volviendo al paper del autor, en el apartado donde se presenta el **coeficiente TDR** explica que éste se basa en otro coeficiente p que **cuanto mayor es, más énfasis se hace en el proceso de explotación**.

Este parámetro p tiene un valor inicial igual a 6 y dado que el autor utiliza por defecto **500 iteraciones** del algoritmo (sin reparar en número de evaluaciones de la función objetivo), como tenemos que cumplir una norma en cuanto al número de evaluaciones, al realizar los cálculos pertinentes nos encontramos con que como mínimo **dispondriamos de poco mas de 3 veces el número de iteraciones por defecto** que propone el autor. Esto es porque en cada iteración se produce una **llamada a la función objetivo por cada individuo de la población** y al trabajar con población 60, para dimensión 10 tenemos cerca de 1700 iteraciones. El parámetro p se modificaría multiplicando el valor inicial por 4 en vez de 3, para optimizar aún mas si cabe la explotación y para que se encuentre en un término medio favorable a ambas dimensiones del problema tratadas (10 y 30).

Establecido este factor, se pasa a definir que el **número de iteraciones** durante las cuales se ejecutará la LS será de **300 iteraciones** (valor propuesto por defecto por el autor del código de CMAES) y que tras una serie de pruebas es el que decidimos mantener para conservar el equilibrio exploración/explotación.

Además hacia falta especificar a partir de que momento se podría aplicar la LS, por lo que se optó por un **umbral** correspondiente al **70 % del total de evaluaciones de la función objetivo**; esto quiere decir que tras gastar un 70 % de evaluaciones de la función objetivo, siempre y cuando se cumpliese esta condición y que la generación de un numero aleatorio en [0,1] fuese menor que 0.25, se ejecutaría la LS.

Definidos todos los factores implicados en esta primera prueba pasamos a establecer una semilla aleatoria con valor igual a 18 para ejecutar el algoritmo básico y el algoritmo memético (con LS) con el objetivo de considerar que tan buenos eran estos parámetros de cara a obtener mejores soluciones. A continuación se muestra una tabla comparativa para cada dimensión, con los valores de la función objetivo obtenidas tanto por el algoritmo básico como por el algoritmo memético, ambos considernado la semilla y una única ejecución por función objetivo.

DIM 10	MVO Básico	MVO Memético	LS Calls
F1	4,04E+04	5,31E+04	28784
F2	2,25E+03	1,02E+03	28784
F3	9,26E+00	3,49E+00	30583
F4	2,29E-01	2,98E-03	28784
F5	2,00E+01	2,02E+01	30583
F6	4,58E+00	3,49E+00	30583
F7	2,54E-01	9,44E-01	30583
F8	1,69E+01	2,19E+01	28784
F9	1,69E+01	1,09E+01	28784
F10	1,66E+02	7,94E+02	30583
F11	7,18E+02	5,43E+02	28784
F12	1,25E-01	5,93E-02	30583
F13	6,71E-02	6,46E-02	30583
F14	5,79E-02	6,55E-02	28784
F15	1,65E+00	2,60E+00	28784
F16	3,12E+00	3,49E+00	28784
F17	1,19E+04	2,14E+03	28784
F18	8,66E+02	1,38E+03	30583
F19	4,93E+00	3,27E+00	30583
F20	9,25E+01	4,81E+01	28784

(a) Comparación de resultados: 20 primeras funciones del CEC 2014, dimensión 10.

DIM 30	MVO Básico	MVO Memético	LS Calls
F1	4,70E+06	2,87E+06	91181
F2	1,22E+04	1,52E+07	91181
F3	1,60E+02	4,30E+01	91181
F4	9,88E+01	7,25E+01	91181
F5	2,00E+01	2,10E+01	91181
F6	1,10E+01	1,10E+01	91181
F7	4,53E-02	1,13E+00	91181
F8	5,85E+01	6,70E+01	91181
F9	8,86E+01	7,10E+01	91181
F10	3,22E+03	2,48E+03	91181
F11	2,46E+03	3,53E+03	91181
F12	1,37E-01	8,10E-01	91181
F13	3,21E-01	5,10E-01	91181
F14	2,22E-01	2,26E-01	91181
F15	5,91E+00	1,86E+01	91181
F16	1,08E+01	1,30E+01	91181
F17	1,67E+04	1,20E+04	91181
F18	1,58E+03	1,13E+05	91181
F19	1,25E+01	9,17E+00	91181
F20	2,53E+02	5,70E+02	91181

(b) Comparación de resultados: 20 primeras funciones del CEC 2014, dimensión 30.

Como se puede apreciar, los resultados de aplicar una Local Search como CMAES sobre una dimensión del problema relativamente pequeña muestra resultados de una calidad considerable. Si miramos los casos donde el algoritmo MVO memético no consigue una mejora frente al algoritmo básico, el valor de la función objetivo no se aleja apenas

del mínimo encontrado por el algoritmo básico, por lo que considerando los resultados se puede concluir que el algoritmo memético es supone una mejora aplicable para una dimensión del problema reducida.

Ocurre lo mismo con el algoritmo memético aplicado a una dimensión mayor (excepto para una única funcion, F2, que se dispara sin explicación aparente) por lo que no es descabellado pensar que la componente aleatoria del algoritmo puede llevarnos a mejores resultados, de media, para 25 ejecuciones. Este será el punto de partida de nuestra segunda y última prueba en cuanto a algoritmos meméticos, descrita a continuación.

3.2. Prueba 2: Ejecución del algoritmo memético y comparación con DE

Tras terminar la primera prueba (y otras que se han descartado para no alargar la memoria, solo ayudaban a ajustar los parámetros utilizados para las posteriores pruebas) vamos a proceder a realizar las mediciones completas de nuestro algoritmo memético; esto es, vamos a utilizar el **algoritmo memético y ejecutar 25 veces cada función** para obtener un valor medio (así como del número de evaluaciones de la función objetivo). Los resultados serán comparados con los que se han obtenido en la sección anterior de la misma forma que se ha hecho hasta ahora, a través de gráficas y tablas de valores de función objetivo. Los tablas de resultados de comparativas deTanabe también se muestran en la memoria.

	DEBin	DEexp	MVO	MVO Memético	LS Calls
F1	0,00E+00	0,00E+00	1,54E+05	2,86E+04	28784
F2	0,00E+00	0,00E+00	4,07E+03	7,27E+03	28784
F3	0,00E+00	0,00E+00	1,43E+01	4,32E+00	30583
F4	2,16E+01	2,19E+01	2,01E+01	2,80E+01	30583
F5	2,02E+01	2,01E+01	2,00E+01	2,03E+01	30583
F6	5,84E-01	1,84E-01	1,55E+00	1,98E+00	28784
F7	3,66E-02	3,58E-02	2,99E-01	7,55E-01	28784
F8	4,35E+00	3,98E-02	1,59E+01	2,26E+01	28784
F9	1,20E+01	8,79E+00	1,66E+01	2,24E+01	30583
F10	5,20E+01	1,15E+01	5,65E+02	7,79E+02	28784
F11	4,44E+02	5,55E+02	6,35E+02	8,39E+02	28784
F12	4,26E-01	4,39E-01	1,58E-01	1,07E-01	30583
F13	1,14E-01	1,12E-01	1,67E-01	1,89E-01	30583
F14	1,76E-01	1,63E-01	1,01E-01	9,96E-02	28784
F15	1,77E+00	1,24E+00	1,23E+00	1,26E+00	28784
F16	2,34E+00	2,35E+00	2,49E+00	2,78E+00	28784
F17	1,64E+01	8,49E+00	3,47E+03	1,84E+03	28784
F18	5,40E-01	5,54E-01	1,27E+04	6,11E+03	28784
F19	3,11E-01	4,53E-01	2,30E+00	2,81E+00	30583
F20	2,04E-01	6,60E-02	1,04E+02	9,47E+01	28784

Figura 3.2: Comparación de resultados fichero Results (DE): 20 primeras funciones del CEC 2014, dimensión 10.

	DEBin	DEexp	MVO	MVO Memético	LS Calls
F1	8,88E+04	2,95E+05	3,19E+06	2,36E+06	91181
F2	2,00E+02	2,00E+02	2,52E+04	1,61E+07	91181
F3	3,00E+02	3,00E+02	4,20E+02	3,38E+02	91181
F4	3,85E+02	4,11E+02	4,86E+02	4,90E+02	91181
F5	5,01E+02	5,00E+02	5,20E+02	5,21E+02	91181
F6	6,03E+02	6,17E+02	6,09E+02	6,12E+02	91181
F7	7,00E+02	7,00E+02	7,00E+02	7,01E+02	91181
F8	8,10E+02	8,00E+02	8,83E+02	9,14E+02	91181
F9	1,02E+03	9,73E+02	9,95E+02	1,02E+03	91181
F10	1,43E+03	9,93E+02	3,82E+03	4,27E+03	91181
F11	6,76E+03	4,31E+03	4,17E+03	4,40E+03	91181
F12	1,20E+03	1,20E+03	1,20E+03	1,20E+03	91181
F13	1,30E+03	1,30E+03	1,30E+03	1,30E+03	91181
F14	1,40E+03	1,40E+03	1,40E+03	1,40E+03	91181
F15	1,51E+03	1,51E+03	1,51E+03	1,52E+03	91181
F16	1,61E+03	1,61E+03	1,61E+03	1,61E+03	91181
F17	4,95E+03	3,93E+03	1,22E+05	6,68E+04	91181
F18	1,82E+03	1,84E+03	1,51E+04	1,46E+05	91181
F19	1,90E+03	1,91E+03	1,91E+03	1,91E+03	91181
F20	2,01E+03	2,03E+03	2,38E+03	2,36E+03	91181

Figura 3.3: Comparación de resultados fichero Results (DE): 20 primeras funciones del CEC 2014, dimensión 30.

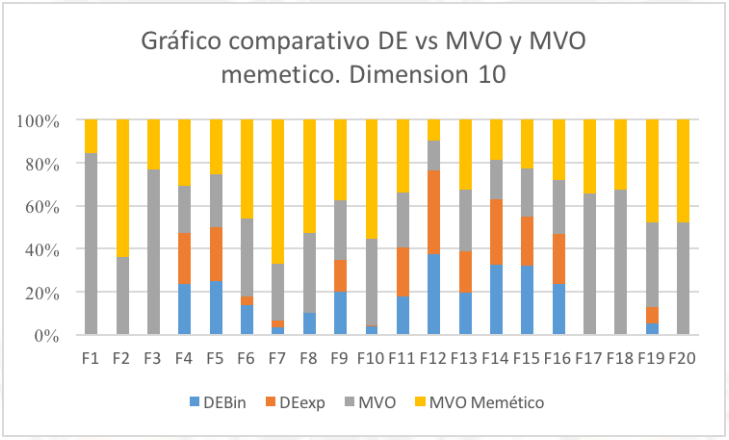

Figura 3.4: Comparación de resultados fichero Results (DE): 20 primeras funciones del CEC 2014, dimensión 10.

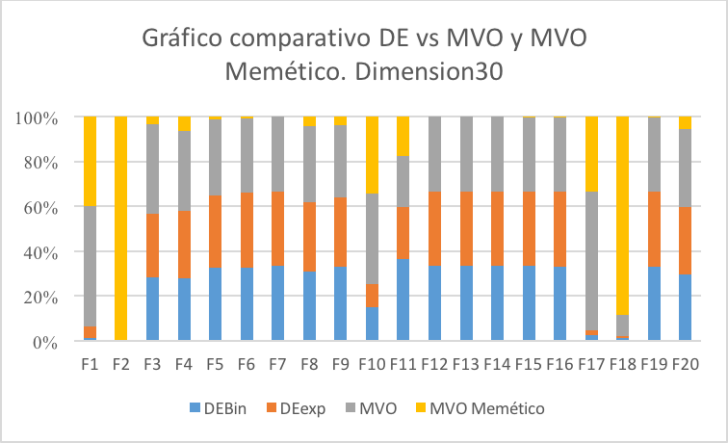

Figura 3.5: Comparación de resultados fichero Results (DE): 20 primeras funciones del CEC 2014, dimensión 30.

Es muy fácil comprobar como la componente aleatoria acaba con cualquier posibilidad de nuestro algoritmo memético de superar al algoritmo básico o incluso a algunos de los DE en dimensión 30. Por otra parte esta misma componente aleatoria es la que permite que se mejoren ligeramente algunos de los resultados para dimensión 10. Si bien no se dispara de forma abrupta en ninguno de los dos casos, tras una primera toma de contacto no se puede concluir con absoluta certeza que un algoritmo memético como el que se ha diseñado pueda ofrecer mejores soluciones en espacios de búsqueda tan grandes, por lo que serán necesarias modificaciones y mejoras posteriores para buscar potenciar las capacidades del algoritmo. Resulta innecesario comparar estos resultados con los del fichero Tanabe pero se ha añadido en el fichero zip junto con la entrega del código fuente para su consulta.

4. Mejoras potenciales de la metaheurística

La última parte de nuestro estudio consiste en proponer y diseñar distintas mejoras potenciales para nuestra metaheurística. Estas mejoras, como bien indica su nombre, tienen como objetivo optimizar los resultados obtenidos tanto por el algoritmo básico y el algoritmo memético, o al menos, proponer distintas vertientes de posibles futuros estudios que permitan encarrilar posteriores mejoras o incluso servir de base para diseñar el algoritmo desde otro enfoque.

Las mejoras que se proponen en esta memoria y que se han implementado en el código han sido testeadas bajo las mismas condiciones que en las anteriores secciones, una primera parte de ejecución con semilla aleatoria para ajustar algunos de los parámetros propuesto y una segunda ejecución que recoge la media de los resultados de 25 ejecuciones por función. El resto de esta sección estará destinado única y exclusivamente a describir y explicar las mejoras propuestas sobre el algoritmo memético (con LS) y posteriormente

se mostrarán y comentarán brevemente sus resultados.

4.1. Mejora 1: diversificación en el proceso de exploración

Las ejecuciones del algoritmo memético y su depuración paso a paso a través de la aplicación de MATLAB mostraba en un gran porcentaje de ejecuciones que los **mecanismos de exploración** de los que estaba dotado nuestro algoritmo básico **no eran lo suficientemente potentes**. Esto fue comprobado de forma experimental a través de varias ejecuciones con distintas funciones cuando tras pasar un gran número de iteraciones, l**a mejor solución obtenida no se optimizaba** tras una cantidad de años luz considerable.

Al observar este comportamiento, optamos por proponer una mejora aplicable al mecanismo de exploración nativo de nuestro algoritmo. Como se ha descrito anteriormente, el **proceso de exploración estaba controlado por los agujeros blancos y negros**, por lo que basándonos en esta misma idea, diseñamos un mecanismo de exploración adaptable basado en la idea de los algoritmos de DE.

Optamos por elegir un mecanismo como el que utiliza un DE Rand/2: elegir a través de nuestro mecanismo de ruleta, **una serie de universos de forma pseudo-aleatoria** (condicionada por el valor de la función objetivo) de forma que a través de un **factor de escalado** se realizase una exploración mucho más amplia del espacio de soluciones. La fórmula que define este comportamiento en la fase de exploración se define a continuación:

$$U(BH_i, j) = U(BH_i, j) + F(U(WH_1, j) - U(WH_2, j) + U(WH_3, j) - U(WH_4, j))$$

Donde BH representa el agujero negro en el universo i-esimo, F el factor de escalado y WH_k el agujero blanco del universo k-esimo elegido de forma aleatoria por el mecanismo de ruleta. De esta forma utilizamos 4 universos distintos aparte del universo actual para calcular el nuevo valor de ese objeto/variable en nuestro universo, dandole una **mayor amplitud de búsqueda a nuestro algoritmo** (al menos, teóricamente) que con la utilización de un único universo. Esta mejora tiene como objetivo principal **aumentar el grado de exploración** de nuestro algoritmo en vistas a un mayor diversificación que permita al algoritmo converger en mejores soluciones. A esta mejora, modelada en el ámbito de la teoría de los multiversos, se le ha llamado **intercambio multiuniversal**.

NOTA: cabe destacar que se ha planteado y diseñado una estrategia similar de diversificación con el mecanismo de DE Current To Best, pero las pruebas inmediatas (tanto con semilla aleatoria como sin semilla) han servido para descartarla como posible opción debido a que al condicionar un universo con otro universo considerado como el mejor pero que puede ser un óptimo local, se pierde toda capacidad de diversificación de nuestras soluciones, dado que el factor de escalado utilizado (el mismo que en el caso de nuestra primera mejora efectiva) influía enormemente en los resultados y el algoritmo no presentaba mejora alguna con respecto a su versión básica.

4.2. Mejora 2: explotación tras N iteraciones sin mejora

Esta mejora viene de la mano con la que se acaba de exponer. El objetivo principal consiste en que, si aún despues de modificar el mecanismo de exploración éste no reporta una optimización de la mejor solución tras un número determinado de iteraciones, se pondrá en marcha nuestra LS para intentar optimizar aquel universo que mejores resultados es capaz de proponer ante el problema, en este caso, una función objetivo.

El diseño y la implementación de esta mejora es sencilla si consideramos una serie de parámetros para medir las distintas situaciones. Requerimos de un parámetro para medir cuantas **iteraciones** han pasado **desde la última actualización** en cuanto al mejor universo, así como un umbral a partir del cual se considera que se aplicará la búsqueda local. Este último parámetro esta condicionada por la dimensión del problema y el tamaño de la población, de tal forma que si la dimension es igual a 10, el umbral tendrá un valor de 50*DIM/10, en cualquier otro caso el valor será de 50*((DIM/10)+1).

Aplicando esta sencilla mejora se ha comprobado de forma experimental como nuestra LS es capaz de ofrecer una vía de escape cuando la aleatoriedad a la que esta sujeta la exploración de nuestro algoritmo (ya modificada con la mejora de deiversificación del apartado anterior) no permite liberarnos de un estancamiento en un posible óptmio local, por lo que a pesar de consumir evaluaciones de la función objetivo, su uso nos lleva a asegurarnos no desperdiciar más evaluaciones de la función objetivo ante una solución o conjunto de éstas que son próximas a un óptimo local. Pasemos con la última mejora.

4.3. Mejora 3: Colapso de universos

En vista de los resultados de las ejecuciones anteriores en cuanto a la capacidad de exploración del algoritmo y el buen balance conseguido entre evaluaciones destinadas a la LS y a procesos que no consideran LS, se propone una nueva mejora con una base sólida en la propia teoría de los multiversos: **el colapso de un universo**. ¿Cómo se modela esta mejora y que ventajas es capaz de ofrecer de cara a obtener mejores soluciones? Lo discutimos a continuación.

La introducción de un nuevo parámetro adaptativo diseñado íntegramente por mi perso-

na, apoyándome en los mismos principios sobre los que se basa el diseño de este algoritmo, permite al algoritmo potenciar aún mas, en teoría, su capacidad de exploración. Este parámetro estará condicionado también por el coeficiente que controla el número de iteraciones transcurridas desde la última actualización de la función objetivo. Se detalla a continuación.

Un universo de baja calidad tiende a **colapsar** cuanto **más tiempo pase sin que el mejor universo haya podido mejorarse**. Dado que durante la fase de explotación propia del algoritmo básico se mejoran el resto de universos, si el mejor universo no ha mejorado en un gran número de años luz, los universos cuyo **ratio de inflación sea mayor** (peor universo) comienzan una etapa de colapso en la que pueden llegar a colapsar de forma parcial o total. El parámetro **UCP (Universe Collapsation Probability)** define la **probabilidad que tiene un universo de baja calidad en colapsar**. La fórmula que define su valor adaptativo es

$$UCP_i = \frac{t_i}{T_{totalC}} * NIR(U_i)$$

donde t_i representa la cantidad de años luz de los totales (T_{totalC}) que han transcurrido sin que el mejor universo de optimice. El NIR(U_i) define el ratio de inflación normalizado del universo i-esimo.

Por lo tanto, ¿como actúa este parámetro adaptativo? Cuanto mayor sea el valor UCP global (sin el NIR del universo i-esimo) mayor probabilidad de colapso de los universos de baja calidad. Este colapso se puede dar en dos fases claramente distinguidas:

1. **Fase 1 - Colapso parcial**: el universo entra en colapso hasta cierto punto, a partir del cual comienza a estabilizarse y a recibir objetos o masa procedente de otro universo (en este caso, el mejor) hasta que completa su estabilización.
2. **Fase 2 - Colapso total**: el colapso del universo es inminente e imparable, por lo que se sustituye por otro universo paralelo generado de forma aleatoria, mientras que a efectos prácticos, el actual desaparece.

Por lo tanto la implementación de ésta mejora (ver código adjunto a la documentación) ha sido simple una vez sentadas las bases de nuestro algoritmo. Tras revisar todos los universos y comprobar si ha habido mejor o no (actualización a un mejor universo), se lanza un número aleatorio que de ser menor que el UCP global abrirá paso a que un conjunto de universos, un 10 % de los peores universos, entren en colapso en función de el coeficiente de colapso parcial PCP que en nuestro caso es igual a 0.6. Por lo tanto un universo tiene un 60 % de probabilidades de colapsar de forma parcial y un 40 % de forma total.

Esta mejora permite una dualidad exploración/explotación con el único objetivo de potenciar ambas vertientes del algoritmo de cara a obtener soluciones mucho mas ajustadas sin necesidad de desperdiciar las mismas en llamadas a una LS que, por términos de aleatoriedad, arroje resultados pobres. A continuación y para concluir con nuestra memoria

se muestran los resultados del algoritmo comparándose con los de DE, esta vez sin los algoritmos del fichero Tanabe pero con los algoritmos participantes en la competición del CEC 2014. Cabe destacar que el umbral a partir del cual se ha de aplicar la LS es de 0.85 frente al 0.7 utilizado en el algoritmo memético, principalmente para mantener el balance entre exploración y explotación.

	DEBin	DEexp	MVO	MVO Memético	LS Calls	MVO Híbrido	LS Calls
F1	0,00E+00	0,00E+00	1,54E+05	2,86E+04	28784	4,21E+04	26985
F2	0,00E+00	0,00E+00	4,07E+03	7,27E+03	28784	1,09E+04	35980
F3	0,00E+00	0,00E+00	1,43E+01	4,32E+00	30583	5,73E+00	26985
F4	2,16E+01	2,19E+01	2,01E+01	2,80E+01	30583	2,55E+01	34181
F5	2,02E+01	2,01E+01	2,00E+01	2,03E+01	30583	2,04E+01	41377
F6	5,84E-01	1,84E-01	1,55E+00	1,98E+00	28784	2,09E+00	35980
F7	3,66E-02	3,58E-02	2,99E-01	7,55E-01	28784	7,93E-01	35980
F8	4,35E+00	3,98E-02	1,59E+01	2,26E+01	28784	2,64E+01	37779
F9	1,20E+01	8,79E+00	1,66E+01	2,24E+01	30583	2,58E+01	41377
F10	5,20E+01	1,15E+01	5,65E+02	7,79E+02	28784	7,02E+02	41377
F11	4,44E+02	5,55E+02	6,35E+02	8,39E+02	28784	6,52E+02	41377
F12	4,26E-01	4,39E-01	1,58E-01	1,07E-01	30583	1,52E-01	39578
F13	1,14E-01	1,12E-01	1,67E-01	1,89E-01	30583	1,63E-01	34181
F14	1,76E-01	1,63E-01	1,01E-01	9,96E-02	28784	1,16E-01	34181
F15	1,77E+00	1,24E+00	1,23E+00	1,26E+00	28784	1,97E+00	37779
F16	2,34E+00	2,35E+00	2,49E+00	2,78E+00	28784	2,92E+00	35980
F17	1,64E+01	8,49E+00	3,47E+03	1,84E+03	28784	2,57E+03	35980
F18	5,40E-01	5,54E-01	1,27E+04	6,11E+03	28784	4,18E+03	39578
F19	3,11E-01	4,53E-01	2,30E+00	2,81E+00	30583	2,81E+00	39578
F20	2,04E-01	6,60E-02	1,04E+02	9,47E+01	28784	1,05E+02	30583

Figura 4.1: Comparación de resultados fichero Results (DE): 20 primeras funciones del CEC 2014, dimensión 10.

	DEBin	DEexp	MVO	MVO Memético	LS Calls	MVO Híbrido	LS Calls
F1	8,88E+04	2,95E+05	3,19E+06	2,36E+06	91181	2,38E+06	43191
F2	2,00E+02	2,00E+02	2,52E+04	1,61E+07	91181	4,51E+06	43191
F3	3,00E+02	3,00E+02	4,20E+02	3,38E+02	91181	3,17E+02	33593
F4	3,85E+02	4,11E+02	4,86E+02	4,90E+02	91181	4,88E+02	33593
F5	5,01E+02	5,00E+02	5,20E+02	5,21E+02	91181	5,21E+02	100779
F6	6,03E+02	6,17E+02	6,09E+02	6,12E+02	91181	6,13E+02	62387
F7	7,00E+02	7,00E+02	7,00E+02	7,01E+02	91181	7,01E+02	33593
F8	8,10E+02	8,00E+02	8,83E+02	9,14E+02	91181	9,07E+02	57588
F9	1,02E+03	9,73E+02	9,95E+02	1,02E+03	91181	1,10E+02	52789
F10	1,43E+03	9,93E+02	3,82E+03	4,27E+03	91181	4,15E+03	47990
F11	6,76E+03	4,31E+03	4,17E+03	4,40E+03	91181	4,31E+03	47990
F12	1,20E+03	1,20E+03	1,20E+03	1,20E+03	91181	1,20E+03	95980
F13	1,30E+03	1,30E+03	1,30E+03	1,30E+03	91181	1,30E+03	95980
F14	1,40E+03	1,40E+03	1,40E+03	1,40E+03	91181	1,40E+03	86382
F15	1,51E+03	1,51E+03	1,51E+03	1,52E+03	91181	1,52E+03	71985
F16	1,61E+03	1,61E+03	1,61E+03	1,61E+03	91181	1,61E+03	91181
F17	4,95E+03	3,93E+03	1,22E+05	6,68E+04	91181	9,33E+04	81583
F18	1,82E+03	1,84E+03	1,51E+04	1,46E+05	91181	2,56E+05	76784
F19	1,90E+03	1,91E+03	1,91E+03	1,91E+03	91181	1,91E+03	71985
F20	2,01E+03	2,03E+03	2,38E+03	2,36E+03	91181	2,35E+03	76784

Figura 4.2: Comparación de resultados fichero Results (DE): 20 primeras funciones del CEC 2014, dimensión 30.

A continuación se muestran las gráficas asociadas a la comparativa entre los DE y el MVO Híbrido (con las 3 mejoras y la LS).

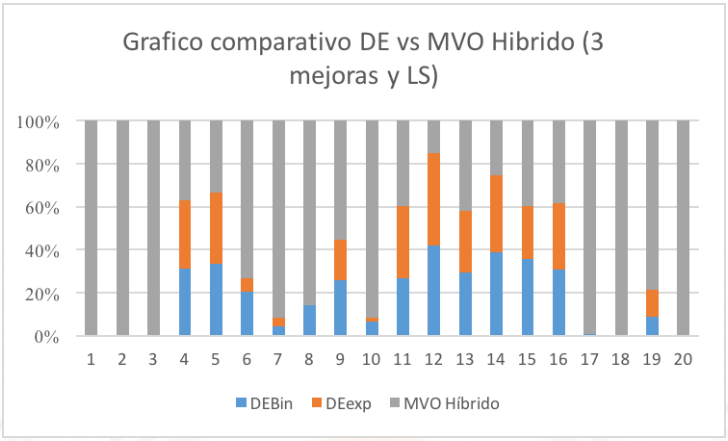

Figura 4.3: Comparación de resultados fichero Results (DE): 20 primeras funciones del CEC 2014, dimensión 10.

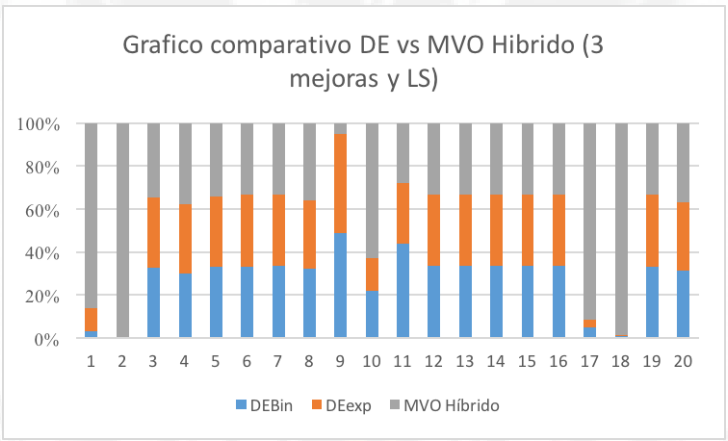

Figura 4.4: Comparación de resultados fichero Results (DE): 20 primeras funciones del CEC 2014, dimensión 30.

Finalmente si se desea consultar la tabla de comparaciones finales con los algoritmos participantes en la competición con nuestro algoritmo MVO Híbrido. Estas tablas no han sido incluidas en la memoria de forma explícita pero se han añadido a los ficheros dentro del zip del proyecto.

5. Conclusiones

El algoritmo MVO es una propuesta nueva dentro de un mundo en donde los algoritmos evolutivos tiene mucho camino recorrido y muchas de éstas técnicas han sido probadas y optimizadas al máximo. Si bien no deja de ser una opción para posibles estudios futuros, no es un algoritmo que sea capaz de hacer frente a los mejores de su ámbito, aún considerando las mejoras que se han aplicado, las cuales parecían prometedoras. El objetivo de este trabajo alternativo consistía en proponer distintas mejoras para una metaheurística y el estudio a fondo de la misma, objetivo que ha quedado plasmado en esta memoria y que pretende ser congruente con el trabajo realizado, proponiendo las tablas y gráficas comparativas tras haber explicado cada una de las distintas cualidades del algoritmo y el porqué de ciertos resultados que se han obtenido tras las ejecuciones, lo que conlleva a un desarrollo del trabajo que pretende ser lo suficientemente completo como para considerarse un estudio a fondo de esta nueva metaheurística.

Referencias

[1] `https://es.mathworks.com/matlabcentral/fileexchange/52898-cma-es-in-matlab`, A structured implementation of Covariance Matrix Adaptation Evolution Strategy (CMA-ES) in MATLAB.

[2] `http://www.alimirjalili.com/MVO.html`, Author Website,.

[3] `http://web.mysites.ntu.edu.sg/epnsugan/PublicSite/Shared%20Documents/Forms/AllItems.aspx`, CEC 2014 Benchmark Function MATLAB Code and Papers.

[4] Multi-verse optimizer: a nature-inspired algorithm for global optimization, Original article published in Neural Comput and Applic.