MOS como Herramienta para la Hibridación de Algoritmos Evolutivos

A. LaTorre, J.M.Peña, J. Fernández, y S. Muelas

Resumen— **El presente trabajo estudia la problemática de la hibridación de algoritmos evolutivos desde la perspectiva de la búsqueda de sinergias entre las distintas aproximaciones (algoritmos) existentes. Para ello se propone MOS (Multiple Offspring Sampling) como un modelo evolutivo que permite, de una manera sistemática, combinar las diferentes aportaciones que estos distintos algoritmos de búsqueda son capaces de proporcionar. En este caso se han seleccionado dos algoritmos evolutivos distintos, Algoritmos Genéticos y Evolución Diferencial, para estudiar su comportamiento al aplicarlos a la resolución de un conjunto de funciones de prueba desarrolladas dentro del marco de la sesión de optimización continua que tuvo lugar en la edición de 2005 del Congreso de Computación Evolutiva, CEC '05 [1].**

Los resultados recogidos en este artículo muestran que la hibridación de distintos algoritmos evolutivos puede tener efectos positivos sobre el proceso de búsqueda, llegando en algunas instancias a obtener mejores resultados que los tres algoritmos de referencia.

Palabras clave— **Multiple Offspring Sampling, Algoritmos Genéticos, Evolución Diferencial, Hibridación de Algoritmos**

I. Introducción

Siempre que nos enfrentamos a un nuevo problema de optimización surge, irremediablemente, la misma pregunta: ¿Cuál es la mejor manera de resolverlo? Para encontrar la respuesta, podemos encomendarnos a nuestros conocimientos en la materia o, mejor aún, a los del conjunto de la comunidad científica, realizando un estudio bibliográfico que nos permita identificar las mejores alternativas existentes y nos ayude a escoger una de ellas. Sin embargo, cualquiera que se haya visto en esta situación se habrá dado cuenta de que esta tarea no es tan sencilla como parece. En la actualidad se publican miles de artículos al año, donde se proponen nuevos algoritmos o variaciones sobre los algoritmos existentes y se prometen mejoras sustanciales con respecto a las herramientas desarrolladas hasta ese momento. Identificar, de entre todas estas alternativas, la que más se adecua a nuestro problema sin ningún conocimiento previo es poco más que una quimera. Probar cada una de ellas, y sus posibles variaciones, requeriría de un tiempo del que, en muchas ocasiones, no se dispone. Además, estudios previos sobre la hibridación de algoritmos evolutivos, a los que haremos referencia en la la sección II, demuestran que, en muchas ocasiones, es posible obtener mejores resultados resolviendo un problema de optimización cuando más de un algoritmo de búsqueda es usado simultáneamente.

En este trabajo vamos a estudiar el comportamiento de MOS en la resolución del conjunto de funciones de prueba propuestas en la sesión especial de optimización continua del congreso CEC '05 [1]. Para ello, combinaremos dos algoritmos de búsqueda de funcionamiento bien distinto. Por un lado, un algoritmo genético, cuyo comportamiento ha sido estudiado en profundidad durante las últimas décadas, y, por otro, la evolución diferencial, de más reciente aparición, pero cuyos interesantes resultados han hecho que acaparen un gran interés en los últimos tiempos. Ambos algoritmos serán descritos, aunque brevemente, ya que no es el objeto de este artículo entrar en los pormenores de cada uno de ellos, en la siguiente sección.

Por último, el artículo se estructura como sigue: la sección II repasa los distintos algoritmos en los que se basa el presente trabajo, así como las aproximaciones existentes en la literatura para la hibridación de algoritmos evolutivos; en la sección III se presentará MOS, y se describirá su estructura y su funcionamiento. En la sección IV se detallarán las contribuciones desarrolladas específicamente para este trabajo, mientras que en la sección V se presentarán tanto la experimentación como los resultados obtenidos en el conjunto de funciones de prueba ya mencionadas. Para finalizar, la sección VI detallará las principales conclusiones extraídas de este estudio.

II. Estado de la cuestión

Los algoritmos genéticos (GAs) son algoritmos de búsqueda basados en población inspirados en la Teoría de la Evolución de Charles Darwin, y que tratan de emular los procesos biológicos en los cuales se inspiran [2]. Para ello, disponen de una población inicial de potenciales soluciones a un problema, de las cuales las más aptas son seleccionadas para reproducirse y obtener así mejores soluciones a dicho problema. Para ello, se dispone de un conjunto de operadores de recombinación que, básicamente, pueden ser de dos tipos. Por un lado, dos o más soluciones pueden ser combinadas por medio de un determinado operador, llamado de cruce, para generar uno o más hijos. Por otro lado, una solución, o individuo, puede sufrir pequeñas modificaciones en forma de mutación que introduzcan cierta diversidad en el conjunto de soluciones. Por último, un mecanismo de reemplazo permite seleccionar, de entre las poblaciones de padres e hijos, los mejores individuos para la siguiente iteración del proceso evolutivo. Siguiendo el símil biológico, cada iteración del proceso recibe el nombre de generación, y éste sigue ejecutándose hasta que se satisface cierto criterio de convergencia.

La evolución diferencial (DE) es otro algoritmo de búsqueda introducido más recientemente y que fue concebido inicialmente para la resolución del problema de ajuste de polinomios de Chebychev [3]. Desde entonces, ha sido satisfactoriamente aplicado a un amplio espectro de problemas de optimización continua [4], [5]. Com-

DATSI, Facultad de Informática, Universidad Politécnica de Madrid. e-mail: {atorre,jmpena,smuelas}@fi.upm.es, jfernandez@laurel.datsi.fi.upm.es.

parte con los GAs su naturaleza estocástica y su funcionamiento basado en poblaciones de soluciones. Difiere, sin embargo, en que, en el caso de la DE, los nuevos individuos son generados mediante un proceso de mutación, mientras que en los GAs el mecanismo principal de reproducción era el operador de cruce. El operador de mutación de la DE añade la diferencia ponderada de dos individuos seleccionados aleatoriamente a un tercer individuo para generar la nueva solución. Posteriormente, esta solución es combinada mediante un operador de cruce con un individuo objetivo para obtener una potencial solución. De entre estas dos soluciones, la obtenida por medio de los operadores de cruce y mutación y el individuo objetivo, se selecciona la de mejor fitness como representante para la siguiente generación.

Desde un punto de vista teórico, existen multitud de partes de un algoritmo evolutivo susceptibles de ser autoreguladas o hibridadas. En [6] el autor establece una clasificación de las distintas estrategias posibles para la adaptación de los parámetros de un algoritmo evolutivo. En primer lugar, distingue entre mecanismos *offline* y *online*. Los primeros intentan encontrar los valores óptimos para los parámetros del algoritmo evolutivo realizando numerosas ejecuciones del algoritmo [7], mientras que los segundos realizan este ajuste al mismo tiempo que el algoritmo está siendo ejecutado. Dentro de este segundo grupo cabe hacer una nueva distinción atendiendo al grado de acoplamiento entre el mecanismo encargado de realizar el susodicho ajuste de parámetros y el propio algoritmo evolutivo. Spears establece aquí tres nuevas categorías: mecanismos fuertemente acoplados, donde es el mismo algoritmo evolutivo el encargado de ajustar sus propios parámetros [8]; medianamente acoplados, donde el algoritmo ajusta sólo en parte sus parámetros; y completamente desacoplados, donde el mecanismo de adaptación es una entidad independiente en comunicación con el algoritmo evolutivo y que evalúa, periódicamente, las condiciones actuales de la búsqueda para adaptar los parámetros del algoritmo evolutivo y obtener el mejor rendimiento posible [9].

El efecto de la codificación de las soluciones de un problema también tiene que ser tenido en cuenta, ya que la elección de un tipo u otro de representación puede hacer que nuestro problema sea más o menos fácil de resolver [10], [11]. Sin embargo, no demasiados estudios se han centrado en este aspecto, aunque sí hay algunos resultados interesantes en la literatura [12], [13].

En el caso concreto de los algoritmos genéticos, existen multitud de estudios donde un mismo algoritmo genético hace uso de distintos operadores, ya sea de cruce o de mutación, para explotar las diferentes características exploratorias de cada uno de ellos y obtener mejores resultados que si los operadores fueran usados de forma independiente [14], [15], [16]. Existen distintas maneras de ajustar la participación de cada uno de los operadores en el proceso de búsqueda conjunto: mediante un controlador *fuzzy* [17], atendiendo a las aportaciones de cada uno de ellos en las últimas generaciones [18] o en base a distintas métricas que evalúen la calidad de las soluciones generadas por cada uno de los operadores [19].

Por último, algunos trabajos proponen algoritmos híbridos donde distintos algoritmos evolutivos completos son usados de manera simultánea, ya sea de forma secuencial [20] o paralela [21], [22].

III. Multiple Offspring Sampling

El teorema del *No Free Lunch* [23] sostiene que *"dos algoritmos cualesquiera son equivalentes si se considera la media de su rendimiento en todos y cada uno de los problemas posibles"* o, lo que es lo mismo, que no puede existir un algoritmo que sea mejor que otro para todos los problemas posibles. Esta afirmación es perfectamente aplicable a los GAs y, por tanto, parece razonable pensar que, ante el desconocimiento a priori acerca del rendimiento de un GA en un problema concreto, es una buena idea hacer uso de distintos mecanismos de optimización simultáneamente para aumentar nuestras posibilidades de éxito. Si a esto le sumamos el hecho de que, como algunos estudios referenciados en la sección anterior confirman [14], [19], la combinación de distintos mecanismos evolutivos puede identificar sinergias existentes entre éstos, es lógico que multitud de trabajos en los últimos años hayan considerado esta posibilidad y la hayan llevado a cabo con un notable éxito.

En [24] se introduce MOS como una herramienta genérica para el desarrollo de algoritmos evolutivos híbridos adaptativos. MOS proporciona la formalización necesaria para diseñar estos algoritmos híbridos, así como las herramientas para identificar y seleccionar el mejor conjunto de parámetros para el problema que se está resolviendo. En este trabajo no entraremos en profundidad en el apartado de formalización de MOS, sino que nos limitaremos a presentar los conceptos más importantes para una buena comprensión de su funcionamiento.

En MOS, el algoritmo principal no maneja un único mecanismo para la producción de nuevos individuos (operadores de recombinación en GAs, modelos probabilísticos en EDAs, operadores de mutación y cruce en DE, etc.) sino que dispone de un conjunto de configuraciones de modelos evolutivos capaces de producir descendencia. Cada una de estas configuraciones recibe el nombre de *técnica* y consiste en (a) un modelo evolutivo concreto, (b) con una representación apropiada, (c) con su conjunto de operadores necesarios (si los utilizara), y (d) configurado con los parámetros apropiados. Cada una de estas técnicas es capaz de producir un subconjunto de nuevos individuos a partir de la población actual. La suma de las subpoblaciones generadas por cada una de las técnicas es igual al tamaño de la población de partida. Inicialmente, todas las técnicas reciben una cuota de participación equitativa en el proceso reproductivo. Dicha participación es ajustada periódicamente (normalmente, cada generación) según una determinada política que habremos de fijar en el comienzo de la ejecución del algoritmo. Este ajuste se realiza mediante lo que recibe el nombre de *función de participación*. Estas funciones pueden ser, desde sencillas asignaciones estáticas de par-

ticipación a cada una de las técnicas o modificaciones lineales de estos valores, a ajustes dinámicos en base a una métrica predefinida. Este último esquema de ajuste de participación recibe el nombre de *ajuste dinámico basado en métrica de calidad*, ya que de lo que se trata es de intentar evaluar la calidad de las soluciones aportadas por cada una de las técnicas reproductivas a la nueva población y, en función de esta calidad, ajustar dinámicamente la participación de cada una de ellas. En este punto, diferentes métricas para la evaluación de la calidad de las aportaciones de una técnica pueden plantearse, en función de lo que entendamos como calidad en cada momento. De este modo, parece evidente que la manera más directa de evaluar la calidad de una técnica es atendiendo al fitness de los individuos que ésta ha generado. Con ello conseguiremos favorecer la generación de individuos por parte de las técnicas que mejores valores de fitness estén consiguiendo en cada momento. Sin embargo, en otros contextos puede ser necesario, por la naturaleza del problema que se está resolviendo o de las técnicas que se están utilizando, garantizar una cierta diversidad de individuos en la población con la que nuestro algoritmo trabaja, por lo que nos puede interesar considerar como medida de calidad la diversidad de los individuos aportados por cada técnica para realizar el ajuste dinámico de participación. Por último, otras medidas pueden ser utilizadas, tales como son la edad de los individuos generados por una u otra técnica o métricas de dificultad de algoritmos, como la Correlación Fitness-Distancia [10] o el Coeficiente de Pendiente Negativa [25], para bonificar la participación de las técnicas para las cuales es menos complicado resolver el problema.

IV. Contribuciones

En este trabajo proponemos un algoritmo híbrido con ajuste dinámico de la participación de las distintas técnicas reproductivas de las que se compone. Este ajuste se realiza analizando la media del fitness de los individuos generados por cada una de las estrategias escogidas. En concreto, se han seleccionado dos tipos de algoritmos de muy distinta concepción, como son los Algoritmos Genéticos y la Evolución Diferencial. A partir de cada uno de estos dos tipos de algoritmos se construyeron dos técnicas. En el caso de las técnicas de Evolución Diferencial, se fijaron sus parámetros a valores recogidos frecuentemente en la literatura como óptimos (tabla I). Para las técnicas basadas en Algoritmos Genéticos se intentó potenciar distintas características exploratorias, haciendo uso de diferentes conjuntos de operadores y valores para los parámetros de cada una de ellas (tabla II).

A partir de estas configuraciones se han establecido cuatro configuraciones de técnicas, seleccionando siempre las dos técnicas de Evolución Diferencial y las dos Genéticas, aunque con distintas probabilidades de mutación en el caso de *GA2*. Además, se probaron dos porcentajes distintos de elitismo: 100 % y 50 %. Todo ello da lugar a cuatro conjuntos de técnicas diferentes, reflejados en la tabla III, y con los que se ha realizado la experimentación.

TABLA I
Parámetros de las técnicas de evolución diferencial

	DE Binomial	DE Exponencial
Selección	Uniforme	Torneo tamaño 2
Cruce	Binomial	Exponencial
F	0.9	0.5
CR	0.5	0.5

TABLA II
Parámetros de las técnicas genéticas

(a) Técnica Genética GA1

Selección	Uniforme
Cruce	Uniforme
Mutación	Uniforme
Prob. Cruce	90 %
Prob. Mutación	10 %

(b) Técnica Genética GA2

Selección	Torneo tamaño 2
Cruce	BLX-α con $\alpha = 0{,}5$
Mutación	Gaussiana
Prob. Cruce	90 %
Prob. Mutación	1 % y 10 %

Otro de los parámetros importantes que hubo que ajustar es el tamaño de población a usar por el algoritmo. Este valor no podía ser muy pequeño debido a que la población es compartida por las cuatro técnicas descritas anteriormente, ni demasiado grande, ya que, al existir un número máximo de evaluaciones, esto podría afectar al rendimiento de las técnicas, especialmente las basadas en DE [26] cuyo rendimiento se ve mermado si el número de generaciones no es el suficiente. Tras una fase de experimentación preliminar, se determinó que el tamaño de población que obtenía mejores resultados, de media, era de 100 individuos.

Finalmente, podemos resaltar un problema surgido a la hora de integrar el algoritmo DE como técnica dentro del esquema de MOS. La evolución diferencial es muy sensible al tipo de elitismo aplicado entre generaciones. En el algoritmo original, tras generar un nuevo individuo, éste es comparado con el individuo de referencia correspon-

TABLA III
Conjuntos de técnicas

MOS1	MOS2	MOS3	MOS4
GA1			
GA2 mut 1 %		GA2 mut 10 %	
DE Exponencial			
DE Binomial			
Eli 100 %	Eli 50 %	Eli 100 %	Eli 50 %

diente, y sólo el mejor de los dos pasa a la siguiente generación. En el modelo de MOS, el elitismo es aplicado entre las poblaciones globales, por lo que, a pesar de que el algoritmo DE descarte a uno de los dos individuos, éste puede ser *rescatado* por el elitismo global, lo cual afecta, como se ha visto en estudios preliminares a este trabajo, muy negativamente en el rendimiento global del algoritmo. Para evitar este problema, la solución de compromiso adoptada ha sido penalizar artificialmente al individuo descartado por la técnica DE, bien sea el individuo de referencia o el recién generado, asignándole el peor valor de fitness posible y evitando así que el elitismo global lo rescate. Esta solución elimina estos problemas en parte aunque, como se verá en el apartado de experimentación, no es suficiente para algunos problemas, por lo que se optará por aumentar la presión evolutiva incrementando la presión selectiva dentro de la técnica DE. Esto, como veremos, conlleva mejoras sustanciales en muchos de los resultados obtenidos.

V. Resultados

Esta sección recoge los resultados obtenidos por las cuatro configuraciones de MOS probadas sobre el subconjunto de funciones de prueba *f6-f25* de la sesión especial de optimización continua del CEC'05 [1]. Las pruebas se realizaron tanto con 10 como 30 dimensiones para cada función, y se llevaron a cabo 25 repeticiones por cada función. El número máximo de evaluaciones se estableció en 100000 para el caso de 10 dimensiones y 300000 para el caso de 30 dimensiones. Las tablas IV y V muestran los resultados de cada configuración en 10 y 30 dimensiones, respectivamente. Ambas tablas recogen la media y la desviación típica obtenidas por cada función y configuración de MOS.

Atendiendo a los resultados presentados en la tabla IV, podemos comprobar que MOS obtiene resultados muy competitivos, mejorando el rendimiento de alguno de los algoritmos de referencia en 18 de las 20 funciones propuestas, y obteniendo el mejor resultado de todos los algoritmos en 3 de esas funciones. Especial atención merece el caso de la función 9, donde 3 de las 4 configuraciones de MOS son capaces de resolver el problema en todas y cada una de las ejecuciones, cosa que no consiguen ninguno de los algoritmos de referencia. Nótese que todas las consideraciones sobre el mejor rendimiento de un algoritmo sobre el otro se hacen únicamente a partir de la información de la que se dispone, que no es más que el error medio de los algoritmos.

Siguiendo con el análisis de los resultados obtenidos en diez dimensiones, vamos a analizar un conjunto de gráficas que muestran la evolución de la participación de las distintas técnicas a lo largo de la ejecución del algoritmo. En la figura 1 podemos observar la evolución de la participación de las diferentes técnicas en la función *f9* de 10 dimensiones. La participación de las técnicas genéticas se limita a tan sólo las 100 primeras generaciones, mientras que la técnica DE que hace uso del operador de cruce binomial ve cómo su participación merma hasta aproximadamente la generación 150 para luego mantenerse estable. A pesar de todo, el algoritmo es capaz de encontrar mejores soluciones, de hecho siempre encuentra el valor óptimo, que si se usara un DE con operador de cruce exponencial únicamente. Esto significa que la provisión de individuos realizada durante las generaciones en que las técnicas genéticas participan de la evolución sumada a la aportación continua de la técnica DE binomial hacen que la técnica DE exponencial, que es la que lleva el peso del algoritmo, mejore sus resultados con respecto a una ejecución independiente. El mismo efecto podemos observarlo en la figura 2, que muestra la evolución de la participación para la función *f15* de 10 dimensiones y la configuración *MOS4*. En este caso, la participación de las técnicas genéticas es más prolongada, por lo que generan individuos más diversos durante más tiempo que en el caso anterior y, además, la participación de la técnica DE binomial es más elevada, por lo que los resultados obtenidos son mucho mejores que en el caso del DE de referencia.

Un análisis distinto podemos hacer de los resultados obtenidos en la función *f25* de 10 dimensiones, donde

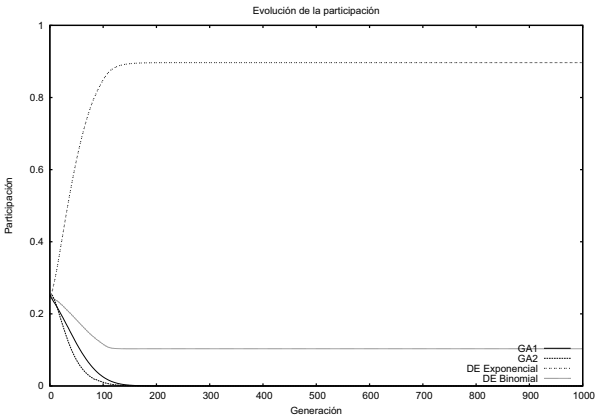

Fig. 1
Evolución de la participación de las técnicas en la función f9 de 10 dimensiones

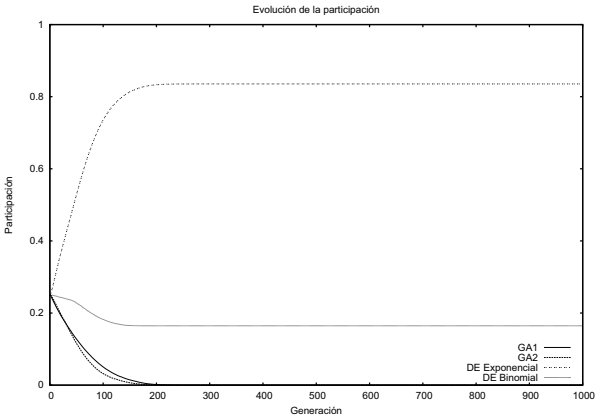

Fig. 2
Evolución de la participación de las técnicas en la función f15 de 10 dimensiones

TABLA IV
RESULTADOS OBTENIDOS EN DIEZ DIMENSIONES

	MOS1	MOS2	MOS3	MOS4
f6	7.51e+00/7.63e+00	4.94e+00/2.00e+00	8.11e+00/1.62e+01	5.19e+00/1.69e+00
f7	4.50e-02/2.35e-02 **D**	3.41e-02/2.65e-02 **D**	3.78e-02/2.14e-02 **D**	5.00e-02/2.87e-02 **D**
f8	2.04e+01/9.33e-02 **D**	2.04e+01/5.80e-02 **D**	2.03e+01/8.34e-02 **D**	2.04e+01/6.92e-02 **D**
f9	3.98e-02/1.95e-01 **GDK**	0.00e+00/0.00e+00 **GDK**	0.00e+00/0.00e+00 **GDK**	0.00e+00/0.00e+00 **GDK**
f10	9.51e+00/3.79e+00 **D**	7.60e+00/3.18e+00 **D**	7.13e+00/3.05e+00 **D**	7.91e+00/3.45e+00 **D**
f11	1.55e+00/1.16e+00 **K**	1.35e+00/1.04e+00 **K**	1.41e+00/7.27e-01 **K**	1.48e+00/1.09e+00 **K**
f12	1.87e+02/4.24e+02	1.02e+02/1.67e+02 **K**	1.94e+02/3.72e+02	3.07e+02/5.35e+02
f13	3.39e-01/9.79e-02 **GDK**	3.60e-01/1.24e-01 **GDK**	3.50e-01/9.56e-02 **GDK**	3.38e-01/1.09e-01 **GDK**
f14	2.69e+00/3.74e-01 **GD**	2.73e+00/3.66e-01 **GD**	2.77e+00/4.34e-01 **GD**	2.82e+00/3.19e-01 **GD**
f15	8.08e+01/1.21e+02 **GDK**	1.19e+02/1.60e+02 **GDK**	9.56e+01/1.35e+02 **GDK**	4.72e+01/7.67e+01 **GDK**
f16	1.05e+02/6.98e+00 **D**	1.02e+02/8.15e+00 **D**	1.08e+02/1.27e+01 **D**	1.03e+02/7.54e+00 **D**
f17	1.04e+02/9.10e+00 **GD**	1.06e+02/8.49e+00 **GD**	1.09e+02/8.85e+00 **GD**	1.04e+02/8.27e+00 **GD**
f18	4.80e+02/2.40e+02 **K**	5.50e+02/2.55e+02 **K**	6.02e+02/2.46e+02 **K**	5.66e+02/2.57e+02 **K**
f19	5.82e+02/2.50e+02 **K**	5.85e+02/2.54e+02 **K**	6.06e+02/2.51e+02 **K**	6.61e+02/2.25e+02 **K**
f20	6.13e+02/2.48e+02 **K**	6.05e+02/2.63e+02 **K**	4.57e+02/2.23e+02 **DK**	5.82e+02/2.62e+02 **K**
f21	6.18e+02/1.74e+02 **K**	5.76e+02/1.75e+02 **K**	6.38e+02/1.89e+02 **K**	6.98e+02/2.09e+02 **K**
f22	7.22e+02/8.67e+01 **G**	7.51e+02/1.44e+01	7.35e+02/7.66e+00	7.02e+02/1.19e+02 **GD**
f23	6.92e+02/1.86e+02 **K**	7.21e+02/2.08e+02 **K**	6.71e+02/1.74e+02 **K**	7.32e+02/1.85e+02 **K**
f24	3.55e+02/7.73e+01 **K**	3.77e+02/5.21e+01 **K**	3.62e+02/7.06e+01 **K**	3.74e+02/6.76e+01 **K**
f25	3.92e+02/1.32e+00 **DK**	3.92e+02/1.24e+00 **DK**	3.92e+02/1.42e+00 **DK**	3.92e+02/1.22e+00 **DK**

G representa que el error medio de MOS es igual o inferior al del algoritmo G-CMA-ES de la sesión CEC'05 [27].
D representa que el error medio de MOS es igual o inferior al del algoritmo DE de la sesión CEC'05.
K representa que el error medio de MOS es igual o inferior al del algoritmo K-PCX de la sesión CEC'05 [28].

TABLA V
RESULTADOS OBTENIDOS EN TREINTA DIMENSIONES

	MOS1	MOS2	MOS3	MOS4
f6	4.04e+03/1.27e+04	2.22e+03/4.03e+03	2.96e+03/5.79e+03	7.12e+03/2.49e+04
f7	2.10e-02/1.40e-02	2.04e-02/1.64e-02	2.63e-02/1.34e-02	2.68e-02/1.53e-02
f8	2.09e+01/4.70e-02 **D**	2.09e+01/5.39e-02 **D**	2.09e+01/8.01e-02 **D**	2.09e+01/6.73e-02 **D**
f9	9.82e+00/3.98e+00 **D**	8.75e+00/2.63e+00 **D**	1.05e+01/3.73e+00 **D**	9.08e+00/3.28e+00 **D**
f10	7.03e+01/1.49e+01 **D**	7.28e+01/1.74e+01 **D**	7.74e+01/1.87e+01 **D**	7.13e+01/1.26e+01 **D**
f11	2.26e+01/2.94e+00 **DK**	2.24e+01/2.96e+00 **DK**	2.21e+01/4.10e+00 **DK**	2.27e+01/2.87e+00 **DK**
f12	2.15e+04/1.40e+04 **G**	2.54e+04/1.53e+04 **G**	1.89e+04/1.34e+04 **G**	2.21e+04/1.43e+04 **G**
f13	4.28e+00/9.63e-01 **K**	3.87e+00/9.18e-01 **K**	4.52e+00/1.13e+00 **K**	4.23e+00/1.04e+00 **K**
f14	1.27e+01/3.13e-01 **GDK**	1.27e+01/2.82e-01 **GDK**	1.26e+01/3.41e-01 **GDK**	1.26e+01/3.75e-01 **GDK**
f15	3.16e+02/8.77e+01 **DK**	3.09e+02/8.39e+01 **DK**	3.49e+02/8.92e+01 **DK**	3.49e+02/9.30e+01 **DK**
f16	1.26e+02/8.60e+01 **D**	1.10e+02/2.61e+01 **D**	1.07e+02/3.37e+01 **D**	1.28e+02/1.02e+02 **D**
f17	1.32e+02/6.75e+01 **GDK**	1.55e+02/9.22e+01 **GDK**	1.22e+02/3.31e+01 **GDK**	1.64e+02/1.17e+02 **GD**
f18	8.34e+02/2.82e+00 **GD**	8.33e+02/1.81e+00 **GD**	8.33e+02/3.38e+00 **GD**	8.33e+02/2.76e+00 **GD**
f19	8.33e+02/1.84e+00 **GD**	8.34e+02/2.03e+00 **GD**	8.34e+02/3.15e+00 **GD**	8.34e+02/2.03e+00 **GD**
f20	8.45e+02/3.26e+00 **GD**	8.43e+02/2.26e+00 **GD**	8.45e+02/2.77e+00 **GD**	8.45e+02/3.19e+00 **GD**
f21	5.44e+02/1.18e+02 **K**	5.58e+02/1.33e+02 **K**	5.73e+02/1.46e+02 **K**	5.87e+02/1.55e+02 **K**
f22	5.43e+02/1.49e+01 **GDK**	5.34e+02/1.25e+01 **GDK**	5.40e+02/2.12e+01 **GDK**	5.34e+02/1.73e+01 **GDK**
f23	5.88e+02/1.24e+02 **K**	5.88e+02/1.23e+02 **K**	6.15e+02/1.43e+02 **K**	6.29e+02/1.51e+02 **K**
f24	2.10e+02/5.88e+00 **GK**	2.10e+02/6.30e+00 **GK**	2.13e+02/3.93e+00 **GK**	2.13e+02/3.85e+00 **GK**
f25	2.13e+02/3.10e+00 **DK**	2.13e+02/3.24e+00 **DK**	2.13e+02/1.47e+00 **DK**	2.12e+02/3.80e+00 **DK**

G representa que el error medio de MOS es igual o inferior al del algoritmo G-CMA-ES de la sesión CEC'05 [27].
D representa que el error medio de MOS es igual o inferior al del algoritmo DE de la sesión CEC'05.
K representa que el error medio de MOS es igual o inferior al del algoritmo K-PCX de la sesión CEC'05 [28].

los resultados de MOS han sido sensiblemente mejores que los del DE de referencia. Si observamos la figura 3 podemos ver cómo, en este caso, la aportación de las técnicas genéticas es testimonial, no colaborando en la producción de individuos más que en las primeras generaciones. A partir de este momento, la participación se la reparten entre las dos técnicas DE donde, si bien existe una de ellas, la exponencial, que adquiere más relevancia, ésta es mucho menor que en funciones anteriores, lo que nos hace pensar que gran culpa del incremento de rendimiento de MOS en esta función tiene que ver con las soluciones aportadas por la otra técnica (la binomial).

Observando la tabla IV, podemos ver que una de las funciones donde peores resultados se obtienen es en la *f6* donde, si bien el error medio no es demasiado elevado, sí está varios órdenes de magnitud por encima de los algoritmos de referencia. Esta vez las gráficas de la evolución de participación no nos son de ayuda, ya que muestran cómo una de las técnicas DE es seleccionada rápidamente como la mejor y se le asigna la mayor parte de la participación. Sin embargo, los valores de la desviación típica nos dan una idea de dónde puede estar el problema. Estos valores son, en proporción a la media, bastante elevados. Si se observa la distribución de los errores de

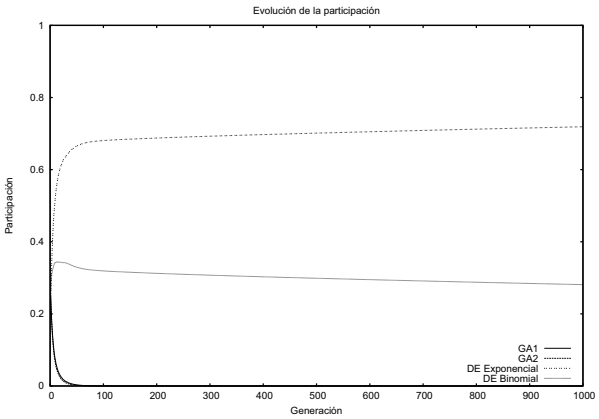

Fig. 3
EVOLUCIÓN DE LA PARTICIPACIÓN DE LAS TÉCNICAS EN LA FUNCIÓN F25 DE 10 DIMENSIONES

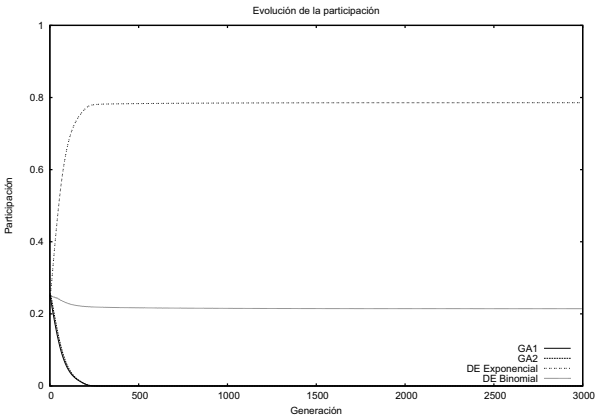

Fig. 4
EVOLUCIÓN DE LA PARTICIPACIÓN DE LAS TÉCNICAS EN LA FUNCIÓN F22 DE 30 DIMENSIONES

las distintas ejecuciones efectuadas, podemos comprobar cómo en algunas de ellas los resultados se encuentran a mucha distancia del óptimo. La conclusión a la que podemos llegar es que, a pesar de que MOS es capaz de detectar cuál de las técnicas empleadas es la que mayor rendimiento está dando, en este problema concreto (donde existe un camino muy estrecho desde el óptimo local al global) esto no es suficiente, y tanto las evaluaciones perdidas con el resto de técnicas como las soluciones generadas por las mismas, degradan el rendimiento de la técnica DE exponencial, provocando, en ocasiones, que ésta no sea capaz de encontrar ese camino y converja a subóptimos muy alejados del óptimo global.

Con respecto a las funciones de 30 dimensiones, los resultados son incluso más competitivos que en el caso de 10 dimensiones, ya que, en este caso, MOS obtiene mejores resultados que alguno de los algoritmos de referencia en 18 de las 20 funciones y, a su vez, obtiene los mejores resultados absolutos en 3 de esas funciones.

Los resultados siguen la misma tendencia que en 10 dimensiones, donde una de las técnicas, la DE exponencial, obtiene grandes cotas de participación en casi todas las funciones. Sin embargo, la aportación de las técnicas genéticas al principio de la ejecución, así como, especialmente, la de la otra técnica DE, la binomial, suponen un estímulo importante en muchas de las funciones y hacen que el comportamiento del algoritmo híbrido mejore con respecto al del DE por sí solo.

Un caso paradigmático de este comportamiento es el obtenido en la función *f22* de 30 dimensiones, donde los resultados de MOS son sensiblemente mejores que los de los algoritmos de referencia. Sin embargo, la participación de las técnicas genéticas se ve reducida a únicamente las primeras 200 generaciones, mientras que la técnica DE binomial ve cómo su participación es reducida levemente al comienzo de la ejecución, para mantenerse posteriormente estable (figura 4). A pesar de ello, la mejora del fitness es considerable con respecto al resto de algoritmos, lo que pone de manifiesto que la contribución de las otras técnicas, por pequeña que ésta haya sido, es determinante en el desempeño del algoritmo híbrido.

Por último, los peores resultados se obtienen, de nuevo, en la función *f6*, donde el comportamiento observado en el problema de 10 dimensiones se ve magnificado, obteniendo resultados de lo más dispar: algunas ejecuciones se acercan al óptimo, mientras que otras se quedan verdaderamente lejos. Como comentamos en la sección IV, uno de los principales problemas que tuvimos que resolver a la hora de introducir una técnica DE dentro del esquema de MOS fue cómo gestionar la presión evolutiva del algoritmo DE original. Por los resultados obtenidos en esta función en concreto, parece que la decisión tomada originalmente no funciona en todos los casos. Es por ello por lo que llevamos a cabo una segunda fase de experimentos, en la que incrementamos esta presión al modificar el mecanismo de selección de los individuos que van a reproducirse mediante las técnicas DE, seleccionando sistemáticamente los mejores individuos de la población global (tantos como su ratio de participación indique) para ser reproducidos mediante esta técnica, en lugar de usar los operadores de selección recogidos en la tabla I.

Las tablas VI y VII muestran las mejoras obtenidas con esta modificación. En 10 dimensiones, conseguimos mejorar los resultados en 10 de las 20 funciones, es decir, justo en la mitad. Sin embargo, no se observan mejoras sustanciales con respecto a los algoritmos de referencia.

En el caso de 30 dimensiones, las mejoras son más importantes, ya que se consiguen mejorar los resultados en 13 de las 20 funciones, un incremento más que considerable. Además, MOS consigue los mejores resultados en 9 de las 20 funciones (las 7 que aparecen en la tabla VII más *f14* y *f17*, problemas en los que no se consigue mejorar con respecto a las anteriores pruebas, pero en los que se mantienen los mejores resultados con respecto a los algoritmos de referencia). De las funciones que no aparecen en la tabla, sólo en una función *f7* se obtienen resultados significativamente peores. En el resto de funciones, el rendimiento es muy similar.

Para finalizar el estudio, cabe destacar que, a pesar de

TABLA VI
MEJORAS OBTENIDAS EN DIEZ DIMENSIONES

	MOS1	MOS2	MOS3	MOS4
f6	2.12e+00/1.85e+00	2.04e+00/1.69e+00	2.93e+00/1.77e+00	2.34e+00/1.97e+00
f7	2.23e+00/2.88e+00	3.00e+00/4.72e+00	1.39e+00/2.16e+00	1.91e+00/3.01e+00
f8	2.03e+01/7.28e-02 **D**	2.04e+01/7.47e-02 **D**	2.04e+01/7.34e-02 **D**	2.03e+01/7.66e-02 **D**
f9	0.00e+00/0.00e+00 **GDK**	0.00e+00/0.00e+00 **GDK**	0.00e+00/0.00e+00 **GDK**	0.00e+00/0.00e+00 **GDK**
f10	7.12e+00/2.51e+00 **D**	7.50e+00/2.69e+00 **D**	6.57e+00/2.51e+00 **D**	7.29e+00/2.25e+00 **D**
f11	5.08e+00/8.30e-01 **K**	5.10e+00/6.13e-01 **K**	4.97e+00/6.51e-01 **K**	5.08e+00/7.07e-01 **K**
f12	7.41e+01/2.25e+02 **K**	4.14e+01/1.06e+02 **K**	8.21e+01/2.43e+02 **K**	6.97e+01/1.34e+02 **K**
f13	2.50e-01/7.45e-02 **GDK**	2.19e-01/6.59e-02 **GDK**	2.51e-01/5.84e-02 **GDK**	2.47e-01/6.13e-02 **GDK**
f14	3.20e+00/1.76e-01 **D**	3.21e+00/1.77e-01 **D**	3.17e+00/1.91e-01 **D**	3.20e+00/2.11e-01 **D**
f15	6.03e+01/1.28e+02 **GDK**	3.82e+01/7.72e+01 **GDK**	7.46e+01/1.43e+02 **GDK**	8.24e+01/1.42e+02 **GDK**
f16	1.02e+02/9.11e+00 **D**	1.03e+02/6.01e+00 **D**	1.01e+02/1.09e+01 **D**	1.01e+02/5.34e+00 **D**
f17	1.08e+02/7.74e+00 **GD**	1.08e+02/7.98e+00 **GD**	1.06e+02/1.00e+01 **GD**	1.08e+02/7.74e+00 **GD**
f18	5.44e+02/2.55e+02 **K**	5.63e+02/2.53e+02 **K**	6.46e+02/2.38e+02 **K**	5.67e+02/2.50e+02 **K**
f19	6.26e+02/2.45e+02 **K**	5.72e+02/2.51e+02 **K**	5.85e+02/2.55e+02 **K**	6.55e+02/2.33e+02 **K**
f20	5.43e+02/2.53e+02 **K**	6.15e+02/2.49e+02 **K**	6.01e+02/2.48e+02 **DK**	6.26e+02/2.45e+02 **K**
f21	5.54e+02/1.77e+02 **K**	5.92e+02/1.82e+02 **K**	5.22e+02/1.64e+02 **K**	5.64e+02/1.73e+02 **K**
f22	7.38e+02/9.60e+01	7.54e+02/6.58e+01	7.30e+02/1.09e+02	7.49e+02/6.42e+01
f23	6.88e+02/1.72e+02 **K**	6.99e+02/1.81e+02 **K**	6.64e+02/1.62e+02 **K**	6.60e+02/1.59e+02 **K**
f24	3.89e+02/2.70e+01 **K**	3.78e+02/5.19e+01 **K**	3.68e+02/6.25e+01 **K**	3.85e+02/3.77e+01 **K**
f25	3.93e+02/1.18e+00 **DK**	3.93e+02/1.02e+00 **DK**	3.92e+02/1.10e+00 **DK**	3.93e+02/8.48e-01 **DK**

G representa que el error medio de MOS es igual o inferior al del algoritmo G-CMA-ES de la sesión CEC'05 [27].
D representa que el error medio de MOS es igual o inferior al del algoritmo DE de la sesión CEC'05.
K representa que el error medio de MOS es igual o inferior al del algoritmo K-PCX de la sesión CEC'05 [28].

TABLA VII
MEJORAS OBTENIDAS EN TREINTA DIMENSIONES

	MOS1	MOS2	MOS3	MOS4
f6	8.80e+01/5.60e+01	7.71e+01/4.91e+01	7.61e+01/7.05e+01	1.00e+02/6.20e+01
f7	7.49e+01/3.97e+01	8.27e+01/3.50e+01	1.15e+02/5.12e+01	1.34e+02/8.28e+01
f8	2.09e+01/5.44e-02 **D**	2.09e+01/4.94e-02 **D**	2.09e+01/5.53e-02 **D**	2.09e+01/4.15e-02 **D**
f9	1.66e-01/3.84e-01 **GDK**	1.21e-01/3.23e-01 **GDK**	1.22e-06/7.24e-06 **GDK**	1.61e-04/1.06e-03 **GDK**
f10	6.50e+01/1.30e+01 **D**	6.57e+01/1.65e+01 **D**	7.48e+01/1.89e+01 **D**	7.08e+01/1.47e+01 **D**
f11	2.40e+01/2.22e+00 **DK**	2.46e+01/2.42e+00 **DK**	2.49e+01/1.93e+00 **DK**	2.41e+01/2.01e+00 **DK**
f12	1.12e+04/7.50e+03 **G**	8.21e+03/6.09e+03 **G**	1.10e+04/7.80e+03 **G**	8.63e+03/6.43e+03 **G**
f13	2.00e+00/3.58e-01 **GDK**	2.10e+00/4.90e-01 **GDK**	2.01e+00/4.18e-01 **GDK**	1.92e+00/3.66e-01 **GDK**
f14	1.28e+01/2.62e-01 **GDK**	1.27e+01/1.75e-02 **GDK**	1.27e+01/2.21e-01 **GDK**	1.27e+01/2.05e-01 **GDK**
f15	3.40e+02/6.93e+01 **DK**	3.16e+02/9.66e+01 **DK**	3.44e+02/9.20e+01 **DK**	3.30e+02/8.77e+01 **DK**
f16	1.21e+02/7.79e+01 **D**	1.00e+02/3.04e+01 **D**	1.02e+02/2.59e+01 **D**	1.19e+02/6.71e+01 **D**
f17	1.72e+02/1.01e+02 **GDK**	1.52e+02/8.78e+01 **GDK**	1.82e+02/1.02e+02 **GD**	1.42e+02/5.17e+01 **GD**
f18	8.21e+02/1.35e+00 **GDK**	8.21e+02/1.58e+00 **GDK**	8.21e+02/1.75e+00 **GDK**	8.21e+02/1.53e+00 **GDK**
f19	8.21e+02/1.42e+00 **GDK**	8.21e+02/1.43e+00 **GDK**	8.21e+02/1.79e+00 **GDK**	8.22e+02/1.93e+00 **GDK**
f20	8.31e+02/1.14e+00 **GDK**	8.31e+02/1.22e+00 **GDK**	8.31e+02/1.55e+00 **GDK**	8.32e+02/1.36e+00 **GD**
f21	5.49e+02/1.23e+02 **K**	5.65e+02/1.39e+02 **K**	5.72e+02/1.44e+02 **K**	6.23e+02/1.71e+02 **K**
f22	5.05e+02/2.78e+00 **GDK**	5.05e+02/2.73e+00 **GDK**	5.05e+02/2.62e+00 **GDK**	5.05e+02/2.63e+00 **GDK**
f23	6.14e+02/1.41e+02 **K**	6.07e+02/1.37e+02 **K**	6.60e+02/1.61e+02 **K**	6.20e+02/1.45e+02 **K**
f24	2.10e+02/3.56e+00 **GK**	2.10e+02/3.55e+00 **GK**	2.11e+02/3.66e+00 **GK**	2.11e+02/5.83e-01 **GK**
f25	2.11e+02/5.25e-01 **GDK**	2.11e+02/6.12e-01 **GDK**	2.11e+02/6.09e-01 **GDK**	2.11e+02/7.62e-01 **GDK**

G representa que el error medio de MOS es igual o inferior al del algoritmo G-CMA-ES de la sesión CEC'05 [27].
D representa que el error medio de MOS es igual o inferior al del algoritmo DE de la sesión CEC'05.
K representa que el error medio de MOS es igual o inferior al del algoritmo K-PCX de la sesión CEC'05 [28].

que las mejoras aportadas por la hibridación de algoritmos suponen un incremento de rendimiento considerable en bastantes de las funciones, éstas son, en general, muy difíciles de resolver por medio de algoritmos evolutivos. En muchos de los casos nunca se llega a encontrar el valor óptimo, especialmente en el caso de las funciones compuestas. En otras, se encuentra esporádicamente, aunque no queda muy claro si es debido a las bondades del proceso de búsqueda que se está efectuando o a simple casualidad en la inicialización o cruce de algunos individuos. Por todo ello, convendría estudiar si este conjunto de funciones es el más apropiado para comparar algoritmos de este tipo o, al menos, incrementar el límite máximo de evaluaciones para dar mayor libertad a la hora de escoger el tamaño de población adecuado o la paralelización del algoritmo, ya que con el número de evaluaciones actual existe muy poca libertad a este respecto.

VI. CONCLUSIONES

Este trabajo propone el uso de MOS como herramienta para la hibridación de algoritmos evolutivos. Para evaluar su rendimiento se ha hecho uso de un subconjunto de las funciones propuestas durante la sesión especial

de optimización continua del congreso CEC'05. Se seleccionaron 4 técnicas evolutivas, dos basadas en algoritmos genéticos y dos basadas en evolución diferencial, y se propusieron distintas combinaciones de las mismas variando algunos de los parámetros que les son propios. Los resultados obtenidos son muy competitivos, obteniendo mejores resultados con respecto a los algoritmos de referencia en varias de las funciones de prueba, especialmente tras aumentar la presión selectiva dentro de las técnicas basadas en evolución diferencial. Sin embargo, y a pesar de todas estas mejoras, en muchas de las funciones de prueba los errores, tanto del algoritmo propuesto como de los algoritmos de referencia, son muy altos, por lo que convendría analizar si este conjunto de funciones es el más apropiado para evaluar la calidad del proceso de búsqueda de este tipo de algoritmos o modificar las condiciones de los ensayos para dar más libertad a la hora de establecer los parámetros de las propuestas (en concreto, incrementar el número máximo de evaluaciones podría contribuir considerablemente).

AGRADECIMIENTOS

Este trabajo ha sido financiado por el Ministerio de Educación y Ciencia (AP-2004-0949 y TIN2007-67148).

REFERENCIAS

[1] P. Suganthan, N. Hansen, J. J. Liang, K. Deb, Y. P. Chen, A. Auger, and S. Tiwari, "Problem definitions and evaluation criteria for the cec 2005 special session on real parameter optimization," Tech. Rep., Nanyang Technological University, 2005.

[2] J.H. Holland, "Adaptation in natural and artificial systems," *University of Michigan Press*, 1975.

[3] R. Storn and K. Price, "Differential evolution- a simple and efficient adaptive scheme for global optimization over continuous spaces," Tech. Rep., 1995.

[4] Hussein A. Abbass, "An evolutionary artificial neural network approach for breast cancer diagnosis," *Artificial Intelligence in Medicine*, vol. 25, pp. 265–281, 2002.

[5] P. Sheth and B. V. Babu, "Optimization of kinetic parameters in pyrolysis of biomass using differential evolution," in *Proc. of International Conference on Neural Network and Genetic Algorithm in Materials Science and Engineering*, January 2008.

[6] W.M. Spears, "Adapting crossover in evolutionary algorithms," in *Proceedings of the Fourth Annual Conference on Evolutionary Programming*, J.R. McDonnell, R.G. Reynolds, and D.B. Fogel, Eds., Cambridge, MA, 1995, pp. 367–384, MIT Press.

[7] J.J. Grefenstette, "Optimization of control parameters for genetic algorithms," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 16, no. 1, pp. 122–128, January 1986.

[8] J.D. Schaffer and A. Morishima, "An adaptive crossover distribution mechanism for genetic algorithms," in *Proceedings of the 2nd International Conference on Genetic Algorithms and their Applications*, J.J. Grefenstette, Ed., Mahwah, NJ, USA, 1987, pp. 36–40, Lawrence Erlbaum Associates.

[9] L. Davis, "Adapting operator probabilities in genetic algorithms," in *Proceedings of the 3rd International Conference on Genetic Algorithms*, San Francisco, CA, USA, 1989, pp. 61–69, Morgan Kaufmann.

[10] T. Jones and S. Forrest, "Fitness distance correlation as a measure of problem difficulty for genetic algorithms," in *Proceedings of the Sixth International Conference on Genetic Algorithms*, Larry Eshelman, Ed., San Francisco, CA, 1995, pp. 184–192, Morgan Kaufmann.

[11] R.A. Caruana and J.D. Schaffer, "Representation of hidden bias: Gray vs. binary coding for genetic algorithms," in *Fifth International Conference on Machine Learning*, 1988, pp. 152–161.

[12] R. Salomon, "The influence of different coding schemes on the computational complexity of genetic algorithms in function optimization," in *Parallel Problem Solving from Nature – PPSN IV*, Hans-Michael Voigt, Werner Ebeling, Ingo Rechenberg, and Hans-Paul Schwefel, Eds., Berlin, 1996, pp. 227–235, Springer.

[13] T. Schnier and X. Yao, "Using multiple representations in evolutionary algorithms," in *Proceedings of the 2000 Congress on Evolutionary Computation*, La Jolla, CA, USA, July 2000, vol. 1, pp. 479–486, IEEE Press.

[14] T.P. Hong, H.S. Wang, and W.C. Chen, "Simultaneously applying multiple mutation operators in genetic algorithms," *Journal of Heuristics*, vol. 6, no. 4, pp. 439–455, September 2000, Kluwer Academic Publishers, Hingham, MA, USA.

[15] D. Thierens, "An adaptive pursuit strategy for allocating operator probabilities," in *GECCO '05: Proceedings of the 2005 Conference on Genetic and Evolutionary Computation*, New York, NY, USA, 2005, pp. 1539–1546, ACM Press.

[16] G. Lin, L. Kang, Y. Chen, B. McKay, and R. Sarker, "A self-adaptive mutations with multi-parent crossover evolutionary algorithm for solving function optimization problems," in *Advances in Computation and Intelligence: Proceedings of the Second International Symposium, ISICA 2007*, L. Kang and Y. Liuand S. Zeng, Eds., Wuhan, China, September 2007, vol. 4683/2007 of *Lectures Notes in Computer Science*, pp. 157–168, Springer-Verlag GmbH.

[17] F. Herrera and M. Lozano, "Adaptation of genetic algorithm parameters based on fuzzy logic controllers," in *Genetic Algorithms and Soft Computing*, F. Herrera and J.L. Verdegay, Eds., pp. 95–125. Physica-Verlag, 1996.

[18] B.A. Julstrom, "What have you done for me lately? adapting operator probabilities in a steady-state genetic algorithm," in *Proceedings of the 6th International Conference on Genetic Algorithms*, L.J. Eshelman, Ed., San Francisco, CA, USA, 1995, pp. 81–87, Morgan Kaufmann.

[19] A. LaTorre, J.M. Peña, V. Robles, and S. Muelas, "Using multiple offspring sampling to guide genetic algorithms to solve permutation problems," in *Proceedings of the 10th Annual Conference on Genetic and Evolutionary Computation, GECCO 2008*, M. Keijzer, Ed., New York, NY, USA, July 2008, pp. 1119–1120, ACM Press.

[20] V. Robles, J.M. Peña, P. Larrañaga, M.S. Pérez, and V. Herves, "Ga-eda: A new hybrid cooperative search evolutionary algorithm," in *Towards a New Evolutionary Computation. Advances in the Estimation of Distribution Algorithms. Series: Studies in Fuzziness and Soft Computing*, J.A. Lozano, P. Larrañaga, I. Inza, and E. Bengotxea, Eds., vol. 192 of *Lecture Notes in Computer Science*, pp. 187–219. Springer-Verlag GmbH, 2004.

[21] V. Bachelet, P. Preux, and E-G. Talbi, "Parallel hybrid metaheuristics: Application to the quadratic assignment problem," in *Proceedings of the Parallel Optimization Colloquium*, 1996, pp. 233–242.

[22] S. Tsutsui and Y. Fujimoto, "Forking genetic algorithm with blocking and shrinking modes (fga)," in *Proceedings of the 5th International Conference on Genetic Algorithms, ICGA '93*, Urbana-Champaign, IL, USA, June 1993, pp. 206–215, Morgan Kaufmann.

[23] D.H. Wolpert and W.G. Macready, "No free lunch theorems for optimization," *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, pp. 67–82, April 1997.

[24] A. LaTorre, J.M. Peña, S. González, V. Robles, and F. Famili, "Breast cancer biomarker selection using multiple offspring sampling," in *Proceedings of the ECML/PKDD 2007 Workshop on Data Mining in Functional Genomics and Proteomics: Current Trends and Future Directions*, Warsaw, Poland, September 2007, Springer Verlag.

[25] R. Poli and L. Vanneschi, "Fitness-proportional negative slope coefficient as a hardness measure for genetic algorithms," in *Proceedings of the 9th Annual Conference on Genetic and Evolutionary Computation (GECCO '07)*, New York, NY, USA, 2007, pp. 1335–1342, ACM Press.

[26] J. Ronkkonen, S. Kukkonen, and K. V. Price, "Real-parameter optimization with differential evolution," in *The 2005 IEEE Congress on Evolutionary Computation.*, 2005, vol. 1, pp. 506–513 Vol.1.

[27] A. Auger and N. Hansen, "A restart cma evolution strategy with increasing population size," in *Proceedings of the IEEE Congress on Evolutionary Computation, CEC 2005*. 2005, pp. 1769–1776, IEEE Press.

[28] A. Sinha, S. Tiwari, and K. Deb, "A population-based, steady-state procedure for real-parameter optimization," in *Proceedings of the IEEE Congress on Evolutionary Computation, CEC 2005*. 2005, vol. 1, pp. 514–521, IEEE Press.