Un algoritmo multiobjetivo basado en búsqueda dispersa

Antonio J. Nebro, Francisco Luna, Enrique Alba, Bernabé Dorronsoro y Juan J. Durillo

Resumen—
Este trabajo analiza una adaptación del algoritmo de búsqueda dispersa para la resolución de problemas de optimización multiobjetivo llamada AbYSS (*Archive-based hYbrid Scatter Search*). AbYSS se considera como un algoritmo híbrido ya que, si bien sigue la estructura de la búsqueda dispersa, utiliza operadores de cruce y mutación típicos en el campo de los algoritmos evolutivos. Se han estudiado diferentes configuraciones de AbYSS para evaluar su capacidad de búsqueda respecto a un conjunto de problemas de prueba. La mejor de estas configuraciones se ha comparado respecto a los dos algoritmos de referencia en el campo, NSGA-II y SPEA2. Los resultados muestran que nuestra propuesta supera a estos dos algoritmos sobre el benchmark utilizado en términos de convergencia y diversidad en los frentes de Pareto obtenidos.

Palabras clave— **búsqueda dispersa, hibridación, optimización multiobjetivo**

I. Introducción

En los últimos años, la optimización de problemas que requieren optimizar más de una función han recibido una gran atención, estando este interés motivado en gran medida por la naturaleza multiobjetivo de los problemas del mundo real [1], [2]. De forma general, la optimización multiobjetivo no se restringe a encontrar una única solución dado un problema de optimización multiobjetivo (MOP, *MultiObjective Problem*), sino en proporcionar un conjunto de soluciones conocido como *soluciones no-dominadas*. Cada solución en este conjunto se dice que es un Óptimo de Pareto, y cuando se representan en el espacio de objetivos, se conocen colectivamente como *Frente de Pareto*. Obtener el frente de Pareto es el principal objetivo en la optimización multiobjetivo. Esto significa que los algoritmos multiobjetivo necesitan explorar grandes porciones del espacio de búsqueda debido a que no se búsca una única solución, sino un conjunto de óptimos de Pareto. A esto podemos añadir que muchos MOPs del mundo real requieren la utilización de métodos computacionalmente complejos para evaluar las funciones objetivo y sus restricciones.

En este contexto, las técnicas deterministas no son aplicables, siendo necesario, por tanto, usar métodos estocásticos. Entre ellos, las metaheurísticas aparecen como una familia de técnicas aproximadas que son muy utilizadas en muchos campos para resolver problemas de optimización; en particular, el uso de algoritmos evolutivos para resolver MOPs ha experimentado un crecimiento significativo en los últimos años, dando lugar a la aparición de una gran variedad de algoritmos, como NSGA-II [3], SPEA2 [4], PAES [5] y muchos otros.

La búsqueda dispersa (*Scatter Search*) [6], [7], [8] es una metaheurística que puede ser considerada como un algoritmo evolutivo en el sentido de que incorpora el concepto de población. No obstante, la búsqueda dispersa evita generalmente el uso de componentes aleatorios, y los operadores típicos de los algoritmos evolutivos como las operaciones de cruce y mutación no se adaptan, teóricamente, a la filosofía de esta técnica. El método se basa en la utilización de una pequeña población, conocida como conjunto de referencia, cuyos elementos se combinan de forma sistemática para la creación de nuevas soluciones. Además, estas nuevas soluciones pueden pasar por una fase de mejora consistente en la aplicación de una búsqueda local. El conjunto de referencia es inicializado a partir de una población inicial, P, compuesta por soluciones aleatorias lo más dispersas posibles, y es actualizado con las soluciones resultantes de la fase de mejora.

Si bien la búsqueda dispersa ha sido utilizada con éxito para resolver una gran variedad de problemas de optimización mono-objetivo [7], su aplicación a problemas multiobjetivo es más bien marginal [9], [10], [11]. No obstante, existen propuestas de algoritmos multiobjetivo basados en esta técnica que son competitivas con el estado del arte en el campo como son NSGA-II y SPEA2. Una de estas propuestas es AbYSS (*Archive-based hYbrid Scatter Search*) [12], que está basado en la plantilla de búsqueda dispersa, pero podemos considerarlo un algoritmo híbrido con algoritmos evolutivos puesto que utiliza operadores de cruce y mutación en su evolución. En este trabajo se presenta un estudio detallado de AbYSS, analizando su capacidad de búsqueda sobre un benchmark concreto (la familia ZDT [13]) en base a los distintos parámetros que definen su comportamiento. Para ello se han utilizado tres métricas diferentes que son estándares en el campo: Generational Distance, Spread e Hypervolume. Posteriormente, la mejor configuración de nuestra propuesta se compara con NSGA-II y SPEA2, los dos algoritmos de referencia en optimización multiobjetivo. Los resultados indican que AbYSS supera a ambos algoritmos proporcionando frentes de Pareto más cercanos a los frente óptimos de los problemas así como con una mejor distribución de soluciones no dominadas.

Dpto. de Lenguajes y Ciencias de la Computación, E.T.S.I. Informática, 29071 Málaga (España). E-mail: {antonio,flv,eat,bernabe,durillo}@lcc.uma.es.

El resto del trabajo se estructura como sigue. En la siguiente sección se presenta nuestra propuesta algorítmica, el algoritmo AbYSS. En la Sección III se presentan el benchmark utilizado, las métricas utilizadas para la comparación de algoritmos, los resultados y su correspondiente análisis. Finalmente, la última sección contiene las conclusiones del artículo y propone algunas líneas de trabajo futuro.

II. AbYSS: Archive-based hYbrid Scatter Search

AbYSS está basado en la plantilla de búsqueda dispersa propuesta en [6] y su aplicación para resolver problemas de optimización monoobjetivo con variables continuas acotadas. La plantilla requiere la definición de cinco métodos: generación de soluciones diversas, mejora, actualización del conjunto de referencia, generación de subconjuntos y combinación de soluciones.

A continuación se describe la plantilla genérica de la búsqueda dispersa para, después, detallar los cinco métodos que la componen. Aquí se hace especial hincapié en los métodos de mejora y actualización del conjunto de referencia, que constituyen la base de AbYSS. Para terminar, se presenta el manejo de soluciones realizado en el archivo externo así como una descripción completa del algoritmo.

A. La Plantilla de Búsqueda Dispersa

El algoritmo de búsqueda dispersa, comienza creando un conjunto inicial de soluciones diversas en la fase de inicialización. Esta fase consiste en generar, de forma iterativa, nuevas soluciones mediante la invocación del método de generación de soluciones diversas; cada nueva solución se pasa al método de mejora, que generalmente consiste en una búsqueda local, y la solución resultante se inserta en la población inicial. Después de esta fase de inicialización, comienza el bucle principal de la búsqueda dispersa.

En el bucle principal, primero se construye el conjunto de referencia a partir del conjunto inicial de soluciones usando el método de actualización del conjunto de referencia. Después, las soluciones en el conjunto de referencia se agrupan sistemáticamente en subconjuntos de dos o más individuos mediante el método de generación de subconjuntos. En el siguiente paso, las soluciones de cada subconjunto son combinadas de alguna manera para producir nuevos individuos. La forma de combinar dichas soluciones la define el método de combinación. El método de mejora se aplica a cada nueva solución generada y, finalmente, se decide cuales de ellas se insertan en el conjunto de referencia. Este bucle se ejecuta hasta que la condición de terminación sea alcanzada (por ejemplo, un número máximo de iteraciones o no se producen nuevos subconjuntos mediante el método de generación de subconjuntos).

Opcionalmente, puede darse un proceso de reinicio. La idea es crear un nuevo conjunto inicial de soluciones que contenga, de partida, las mejores soluciones que se encuentran en el conjunto de referencia, mientras que los restantes individuos son generados mediante el método de generación de soluciones diversas.

B. Métodos de la Búsqueda Dispersa

Para la descripción del algoritmo, sean P y $RefSet$ el conjunto inicial y el conjunto de referencia, respectivamente.

B.1 Método de Generación de Soluciones Diversas

Este método es básicamente el mismo propuesto en [8] y consiste en dividir el rango de cada variable de decisión en un número de subrangos de igual tamaño; entonces, cada nueva solución se obtiene en dos pasos. Primero, se selecciona un subrango de forma aleatoria con una probabilidad inversa y uniformemente proporcional al número de veces que dicho subrango ha sido seleccionado; y, en segundo lugar, se elige aleatoriamente un valor dentro del subrango seleccionado.

B.2 Método de Mejora

La idea tras este método es usar un algoritmo de búsqueda local para mejorar las nuevas soluciones obtenidas mediante el método de generación de soluciones diversas y método de combinación de soluciones. En [8] se usa el método *simplex*; aquí también se han realizado pruebas con la utilización de dicho método, pero también se ha analizado una mejora basada en un (1 + 1) EA (Evolutionary Algorithm). Éste utiliza un operador de mutación usado como perturbación junto con un test de dominancia de Pareto. No seguimos, por tanto, la filosofía de la búsqueda dispersa sobre evitar el uso de operadores estocásticos. Esta decisición está justificada debida a la simplicidad del método de mejora resultante y los beneficios de usar un operador que ha mostrado un buen comportamiento en algoritmos evolutivos.

El método de mejora toma un individuo como parámetro, que se muta repetidamente con el objeto de obtener mejores soluciones. El termino "mejor" se define aquí de una manera similar al enfoque usado en NSGA-II [3]. Un test de violación de restricciones comprueba cuando dos individuos son factibles o no. Si uno de ellos es fáctible y el otro no, o si ambos no son factibles pero uno de ellos viola las restricciones en menor grado que el otro, el test devuelve el vencedor. En otro caso, un test de dominancia se aplica para decidir cuando uno de los individuos domina al otro. Si el original gana, el mutado es descartado; si el mutado gana, éste reemplaza al original; finalmente si ambos son no-dominados, el original se envía al archivo externo y el mutado pasa a ser el original.

B.3 Método de Actualización del Conjunto de Referencia

El conjunto de referencia es una colección que contiene tanto soluciones de alta calidad como solucio-

nes diversas que son utilizadas para generar otras nuevas. El $RefSet$ está compuesto a su vez por dos subconjuntos, $RefSet_1$ y $RefSet_2$, de tamaño p y q, respectivamente. El primero de ellos contiene las soluciones de mejor calidad en P, mientras el segundo de ellos incluye aquellas soluciones que incorporan diversidad al conjunto. Siguiendo el esquema propuesto en [10], AbYSS construye el $RefSet_2$ seleccionando aquellos individuos de P cuya mínima distancia euclídea al $RefSet_1$ sea máxima. Sin embargo, para crear el conjunto $RefSet_1$ hemos tenido que redefinir el concepto de "mejor individuo".

Como se dijo anteriormente, este método se usa además para actualizar el conjunto de referencia con las nuevas soluciones obtenidas durante el bucle principal de la búsqueda dispersa. Así, para seleccionar los p mejores individuos de P, AbYSS utiliza la estrategia de SPEA2, es decir, a los individuos en P se les asigna un valor de fitness que será la suma del strength raw fitness y un valor de estimación de densidad de soluciones [4].

Una vez construido el conjunto de referencia, sus soluciones se combinan para generar nuevos individuos que, al acabar dicha fase, son pasados al procedimiento de mejora. Después, hay que comprobar si estas nuevas soluciones han de ser incluidas en el conjunto de referencia. Así , una nueva solución se convierte en miembro del conjunto de referencia si se cumple alguna de las siguientes condiciones:

- El nuevo individuo es mejor respecto al valor de la función objetivo que el peor individuo del $RefSet_1$.
- El nuevo individuo tiene un mejor valor de distancia al conjunto de referencia que el individuo con peor valor de distancia en $RefSet_2$.

Si bien la segunda condición es válida tanto para mono como para multiobjetivo, la primera de las condiciones supone redefinir de nuevo el concepto de mejor individuo. Aquí no podemos hacer uso de un procedimiento de ranking debido a que el tamaño de esta población es generalmente pequeño (típicamente, el $RefSet$ está formado en total por unos 20 individuos o menos). Nuestro enfoque es comparar cada nueva solución i con todos los individuos en $RefSet_1$ usando un test de dominancia. Así, cuando un individuo no es dominado por ninguna de las soluciones en $RefSet_1$, este se inserta en el conjunto de referencia sólo si este no está ya completo. Esto significa que el nuevo individuo debe dominar al menos una solución en $RefSet_1$ para poder insertarlo. Si esta condición no se cumple, el individuo se envía al archivo externo.

B.4 Método de generación de Subconjuntos

De acuerdo con la plantilla de búsqueda dispersa, este método genera subconjuntos de individuos que se usarán para la creación de nuevas soluciones mediante el método de combinación de soluciones. Es posible generar distintos tipos de subconjuntos. Generalmente, este método genera todos los pares posibles de soluciones del conjunto de referencia. No obstante, en AbYSS este método genera, por un lado, pares de individuos que pertenecen a $RefSet_1$ y , por otro lado, pares de individuos de $RefSet_2$. El estudio experimental que se presenta más abajo muestra que la combinación de elementos los dos subconjuntos provoca en una pobre convergencia del algoritmo. La razón se debe a que la combinación de individuos de ambos subconjuntos aumenta la capacidad de exploración del algoritmo que hace que AbYSS requiera un mayor esfuerzo computacional para converger frentes adecuados.

B.5 Método de combinación de Soluciones

En la metodología original de búsqueda dispersa, este método computa combinaciones lineales de soluciones del conjunto de referencia. No obstante, en este estudio tratamos de demostrar que, usando esta estrategia, los resultados son competitivos en muchos problemas, pero el algoritmo falla cuando intenta resolver problemas más complejos. Nuestra propuesta consiste en usar un operador de cruce SBX (Simulated Binary Crossover) [14] que hace a AbYSS más robusto.

C. Gestión del Archivo Externo

El principal objetivo del archivo externo (o repositorio) es mantener un registro historíco de las soluciones no dominadas encontradas durante la búsqueda, intentando, a la vez, mantener aquellas que producen una mejor distribución sobre frente de Pareto. La cuestión clave en la gestión del archivo es decidir cuándo una solución se debe añadir al archivo.

Cada vez que llega una solución no dominada procedente del bucle interno del método de búsqueda dispersa, se compara una por una con las que ya pertenecen al archivo. Si la nueva solución es dominada por algún individuo en el mismo (es decir la solución está dominada por el archivo), entonces se descarta; en otro caso, la solución es almacenada en el archivo y se eliminan aquellas que están ahora dominadas. Además, si el archivo alcanza su tamaño máximo después de añadir una nueva solución, hay que decidir qué solución debe ser suprimida. Existen diversas estrategias propuestas en la literatura, como el grid adaptativo utilizado en PAES [5] y MOPSO [15]. Aquí, sin embargo, utilizamos la distancia de crowding del NSGA-II [3].

Es importante destacar en este punto que podríamos haber usado el estimador de densidad de SPEA2, como hicimos para el método de actualización del conjunto de referencia. No obstante, hemos decidido utilizar dos estimadores de densidad distintos intentando conseguir una mejor distribución de soluciones no dominadas en el frente de Pareto. El fundamento de esta decisión es que así las soluciones no dominadas han de pasar dos filtros: en primer lugar, las soluciones no pertenecen a la región más poblada de acuerdo con la distancia de crowding y, en segundo lugar, se obtienen a partir de las mejores

soluciones del conjunto inicial teniendo en cuenta el estimador de densidad de SPEA2. Resultados experimentales demuestran que la combinación de ambos estimadores obtienen mejor diversidad en los frentes que usar de sólo uno.

D. Descripción de AbYSS

Una vez que se han definido los cinco métodos de la búsqueda dispersa así como la gestión del archivo de soluciones no dominadas, podemos ofrecer una descripción completa del funcionamiento de nuestra propuesta.

Inicialmente, el método de generación de soluciones diversas es invocado para generar s soluciones iniciales, cada una de las cuales se pasa al método de mejora. Como resultado se obtiene el conjunto inicial P. Tras esta fase inicial, se realiza un determinado número de iteraciones. En cada iteración, se crea el conjunto de referencia, se invoca al método de generación de subconjuntos, y se ejecuta el bucle principal de la búsqueda dispersa hasta que el método de generación de subconjuntos no genere nuevos subconjuntos de soluciones. Después, hay una fase de reinicio que consiste en tres pasos. Primero, los individuos en $RefSet_1$ se insertan en P; segundo, los n mejores individuos del archivo externo, de acuerdo a su distancia de crowding, se mueven también a P; y, tercero, se usan el método de generación de soluciones diversas y método de mejora para producir nuevas soluciones hasta completar P.

La idea de mover n individuos desde el archivo al conjunto inicial es intensificar la búsqueda del algoritmo sobre el frente de Pareto encontrado hasta ese momento. El grado de intensificación puede variar dependiendo del número de individuos elegidos. AbYSS usa un valor de n que es el mínimo entre el tamaño del archivo y la mitad del tamaño de P. La condición de parada del algoritmo es la realización de un número máximo de evaluaciones.

III. Experimentos

En esta sección se presentan los problemas multiobjetivo utilizados en los experimentos, las métricas usadas para medir la calidad de los frentes de Pareto que obtienen los algoritmos, el análisis del comportamiento de AbYSS respecto a los diferentes parámetros que definen el tipo de búsqueda del algoritmo y, finalmente, una comparativa entre la mejor configuración de AbYSS y NSGA-II y SPEA2.

A. Problemas de Prueba

El conjunto de MOPs que se van a usar a lo largo de este trabajo para evaluar los diferentes algoritmos es la familia ZDT [13], ampliamente utilizada en estudios anteriores en el campo: los problemas ZDT1 (convexo), ZDT2 (no convexo), ZDT3 (no convexo, desconectado), ZDT4 (no convexo, multimodal), and ZDT6 (no convexo, no uniformemente espaciado). Hemos considerado que este conjunto de problemas es lo suficientemente amplio y diverso para el estudio que se pretende realizar en este trabajo.

B. Métricas de Rendimiento

A la hora de evaluar el rendimiento de algoritmos multiobjetivo se suelen tener en cuenta dos aspectos: minimizar la distancia del frente de Pareto obtenido por el algoritmo al frente de Pareto exacto del problema y maximizar la extensión de soluciones sobre el frente de forma que la distribución sea lo más uniforme posible. En este sentido, las métricas usadas en el campo se pueden agrupar entre aquellas que evalúan la cercanía a los frentes exactos, las que miden la diversidad de las soluciones no dominadas obtenidas, o ambas [2]. Aquí hemos usado una de cada tipo:

▪ **Generational Distance** Esta métrica la propusieron Van Veldhuizen and Lamont [16] para medir la distancia existente entre el conjunto de soluciones no dominadas encontrado y el conjunto de óptimos de Pareto. Se define como sigue:

$$GD = \frac{\sqrt{\sum_{i=1}^{n} d_i^2}}{n} \ , \tag{1}$$

donde n es el número de elementos del conjunto de soluciones no dominadas y d_i es la distancia euclídea (medida en el espacio de objetivos) entre cada una de éstas soluciones y el miembro más cercano del frente de Pareto óptimo. Es necesario normalizar el conjunto de soluciones para obtener resultados fiables.

▪ **Spread** Aquí se ha empleado una extensión de la métrica Spread propuesta originalmente por Deb *et al.* [3]. En lugar de utilizar la distancia entre dos soluciones consecutivas, la nueva métrica usa la distancia de cada punto a su vecino más cercano. Esta modificación está basada en la métrica descrita en [17]:

$$\Delta = \frac{\sum_{i=1}^{m} d(e_i, S) + \sum_{X \in S} \left| d(X, S) - \bar{d} \right|}{\sum_{i=1}^{m} d(e_i, S) + |S| * \bar{d}} \tag{2}$$

donde S es un conjunto de soluciones, S^* es el conjunto de óptimos de Pareto, $(e_1, ..., e_m)$ son las m soluciones extremas de S^*, m es el número de objetivos y

$$d(X, S) = \min_{Y \in S, Y \neq X} ||F(X) - F(Y)||^2 , \tag{3}$$

$$\bar{d} = \frac{1}{|S^*|} \sum_{X \in S^*} d(X, S). \tag{4}$$

Idealmente, $\Delta(X, S) = 0$ para aquellos conjuntos de soluciones con una distribución óptima sobre el frente de Pareto. Esta métrica también requiere normalización previa para su aplicación.

▪ **Hypervolume** Esta métrica calcula el volumen (en el espacio de objetivos) cubierto por miembros del un conjunto dado, Q, de soluciones no dominadas para problemas donde todos los objetivos han de ser minimizados. Matemáticamente, para cada $i \in Q$ se

construye un hipercubo v_i con un punto de referencia W y la solución i que definen la diagonal del mismo. El punto W se puede obtener simplemente con los peores valores de las funciones objetivo. Entonces, la unión de todos los hipercubos es lo que define el hipervolumen (HV):

$$HV = volume\left(\bigcup_{i=1}^{|Q|} v_i\right). \quad (5)$$

Los algoritmos que alcanzan mayores valores para HV son mejores. Como ocurría con las dos métricas anteriores, es necesario normalizar las soluciones no dominadas puesto que HV depende del escalado de los valores de la función objetivo.

C. Resultados

La capacidad de búsqueda de la mayoría de las metaheurísticas está muy influenciada por su configuración: inicializaciones, operadores, probabilidades de aplicación, etc. Como aparece en la descripción de AbYSS, existen multitud de parámetros que han quedado abiertos y que nos van a permitir definir su comportamiento. Si bien no pretendemos hacer un análisis extensivo de la parametrización, si vamos a estudiar aquellos que son más importantes. La intención es alcanzar un algoritmo lo más eficiente y robusto posible, siempre respecto al benchmark utilizado (Sección III-A). En concreto, se han estudiado los siguientes aspectos de AbYSS:

- Utilización del método simplex en el método de mejora en lugar del (1+1) EA.
- Aplicar el operador de cruce SBX como método de combinación de soluciones en lugar de las combinaciones lineales de soluciones.
- Influencia del tamaño del conjunto inicial P.
- Generar todos las posibles combinaciones de pares de individuos en el método de generación de subconjuntos.
- Número de iteraciones en el método de mejora.

Antes de pasar a la descripción de los experimentos, queremos detallar el método simplex que ha sido desarrollado. Partiendo del método original de Nelder and Meax [18], se ha tenido que adaptar para resolver problemas multiobjetivo.

Hemos tenido que adaptar el método original de Nelder and Meax [18] para resolver problemas de optimización multiobjetivo. La estrategia adoptada consiste en comparar soluciones usando tests de dominancia, considerando como mejores aquellos nuevos individuos que se generan, tanto si dominan como si son no dominados respecto al actual. Además, en lugar de generar soluciones aleatorias cuando generamos el simplex, éste se construye a partir de soluciones del archivo. Esto permite reducir notablemente el número de evaluaciones que requeriría el método.

A continuación se describen los diferentes experimentos realizados con AbYSS:

Experimento 1 : Se usa una configuración de AbYSS típica en el campo de la búsqueda dispersa: el método de mejora es el simplex descrito anteriormente (5 iteraciones), combinación lineal de soluciones para crear nuevas soluciones, el tamaño de P es 100 y el método de generación de subconjuntos crea pares de soluciones que pertenecen tanto al $RefSet_1$ como al $RefSet_2$. El tamaño de ambos subconjuntos es 10.

Experimento 2 : Aquí reemplazamos el método simplex en la mejora e incorporamos el (1+1) EA. Se usa el operador de mutación polinomial [14] con índice de distribución 20. El número de iteraciones del método de mejora es 1.

Experimento 3 : Se mantiene la misma configuración que en el Experimento 2, a diferencia del tamaño de P, que pasa de 100 a 20 individuos. A medida que la estrategia de reinicialización comienza a funcionar, este cambio en el tamaño de P hace que 10 soluciones procedan del $RefSet_1$ y otras 10 sean del archivo externo, por lo que no se genera ninguna solución aleatoria nueva. Esto permite intensificar la búsqueda del algoritmo en zonas muy prometedoras del espacio de búsqueda.

Experimento 4 : AbYSS se configura como en el experimento anterior pero ahora el método de generación de subconjuntos crea pares de soluciones que pertenecen sólo al $RefSet_1$ o al $RefSet_2$. Con esto estamos intensificando aún más la búsqueda de dos formas. Explícitamente, reduciendo el número de combinaciones con soluciones diversas procedentes del $RefSet_2$ y también implícitamente, ya que cuanto menor es el número de combinaciones posibles, más corto es el bucle interno de la búsqueda dispersa y, por tanto, mayor el número de fases de reinicialización con el consiguiente feedback de soluciones no dominadas del archivo.

Experimento 5 : Existen trabajos de búsqueda dispersa monobjetivo en los que operadores estocásticos han logrado mejores resultados que el método "tradicional" de combinación lineal de soluciones [19]. Es por esto por lo que repetimos el Experimento 4 utilizando ahora el operador de cruce SBX con índice de distribución 20.

Experimento 6 : En este último experimento mantenemos la configuración del anterior a excepción del número de iteraciones del método de mejora, que pasa de 1 a 5.

En cada uno de lo experimentos la condición de parada de AbYSS es evaluar 25.000 soluciones. Se han realizado 100 ejecuciones independientes de cada uno. Los resultados obtenidos se muestran en la Tabla I, que incluye la mediana, $\tilde{x}$, y el rango intercuartílico, IQR, como medidas de localización y dispersión estadística, respectivamente. Ya que estamos trabajando con algoritmos estocásticos y queremos dar confianza estadística a los resultados, hemos realizado el siguiente análisis estadístico. Primero aplicamos un test de Kolmogorov-Smirnov para determinar si los valores obtenidos siguen una distribución normal (gaussiana). Si la siguen, utilizamos un test ANOVA, y si no, utilizamos un test de Kruskal-

TABLA I

RESULTADOS DE LAS DIFERENTES CONFIGURACIONES DE ABYSS.

	Generational Distance (GD)						
Problem	Experiment 1 $\tilde{x}_{IQR}$	Experiment 2 $\tilde{x}_{IQR}$	Experiment 3 $\tilde{x}_{IQR}$	Experiment 4 $\tilde{x}_{IQR}$	Experiment 5 $\tilde{x}_{IQR}$	Experiment 6 $\tilde{x}_{IQR}$	
ZDT1	1,546e-4 $_{9,81e-5}$	2,812e-2 $_{6,25e-2}$	5,309e-2 $_{1,06e-1}$	4,937e-2 $_{2,54e-2}$	1,876e-4 $_{3,86e-5}$	2,162e-4 $_{5,15e-5}$	+
ZDT2	1,745e+0 $_{7,32e-1}$	5,649e-5 $_{4,22e-4}$	5,095e-5 $_{6,81e-4}$	2,838e-2 $_{3,51e-2}$	1,020e-4 $_{3,82e-5}$	1,448e-4 $_{6,83e-5}$	+
ZDT3	7,396e-3 $_{1,43e-2}$	3,421e-2 $_{4,68e-2}$	3,826e-2 $_{4,67e-2}$	4,681e-2 $_{1,63e-2}$	1,964e-4 $_{2,17e-5}$	1,978e-4 $_{2,47e-5}$	+
ZDT4	1,403e+1 $_{8,60e+0}$	8,061e-1 $_{1,03e+0}$	3,531e-1 $_{6,92e-1}$	5,674e-2 $_{4,36e-2}$	5,409e-4 $_{3,56e-4}$	7,251e-4 $_{5,93e-4}$	+
ZDT6	1,108e-1 $_{9,45e-2}$	3,484e-3 $_{2,29e-2}$	3,114e-3 $_{1,37e-2}$	5,165e-2 $_{4,32e-2}$	5,494e-4 $_{2,31e-5}$	5,581e-4 $_{2,65e-5}$	+
	Spread (Δ)						
Problem	Experiment 1 $\tilde{x}_{IQR}$	Experiment 2 $\tilde{x}_{IQR}$	Experiment 3 $\tilde{x}_{IQR}$	Experiment 4 $\tilde{x}_{IQR}$	Experiment 5 $\tilde{x}_{IQR}$	Experiment 6 $\tilde{x}_{IQR}$	
ZDT1	1,863e-1 $_{1,02e-1}$	8,882e-1 $_{2,06e-1}$	9,014e-1 $_{1,68e-1}$	7,095e-1 $_{7,56e-2}$	1,136e-1 $_{2,06e-2}$	1,316e-1 $_{2,50e-2}$	+
ZDT2	5,611e+3 $_{5,56e+0}$	1,010e+0 $_{7,28e-2}$	1,005e+0 $_{4,68e-2}$	7,776e-1 $_{1,67e-1}$	1,136e-1 $_{2,05e-2}$	1,379e-1 $_{3,03e-2}$	+
ZDT3	8,327e-1 $_{2,46e-1}$	8,932e-1 $_{1,40e-1}$	9,166e-1 $_{1,23e-1}$	7,882e-1 $_{6,14e-2}$	1,322e-1 $_{2,11e-1}$	1,806e-1 $_{2,06e-1}$	+
ZDT4	9,943e-1 $_{1,34e-1}$	9,906e-1 $_{4,78e-2}$	9,988e-1 $_{5,54e-2}$	8,507e-1 $_{2,38e-1}$	1,241e-1 $_{3,26e-2}$	1,525e-1 $_{4,17e-2}$	+
ZDT6	8,368e-1 $_{7,15e-1}$	1,542e-1 $_{6,22e-1}$	1,424e-1 $_{3,46e-1}$	6,322e-1 $_{6,16e-1}$	9,182e-2 $_{1,67e-2}$	1,133e-1 $_{1,75e-2}$	+
	Hypervolume (HV)						
Problem	Experiment 1 $\tilde{x}_{IQR}$	Experiment 2 $\tilde{x}_{IQR}$	Experiment 3 $\tilde{x}_{IQR}$	Experiment 4 $\tilde{x}_{IQR}$	Experiment 5 $\tilde{x}_{IQR}$	Experiment 6 $\tilde{x}_{IQR}$	
ZDT1	6,615e-1 $_{6,18e-4}$	3,212e-1 $_{4,04e-1}$	1,113e-1 $_{3,91e-1}$	1,828e-1 $_{1,63e-1}$	6,614e-1 $_{3,45e-4}$	6,607e-1 $_{4,66e-4}$	+
ZDT2	0,000e+0 $_{0,00e+0}$	3,994e-2 $_{4,65e-2}$	3,111e-2 $_{3,90e-2}$	1,110e-1 $_{1,61e-1}$	3,282e-1 $_{3,43e-4}$	3,277e-1 $_{5,36e-4}$	+
ZDT3	4,061e-1 $_{1,59e-1}$	1,504e-1 $_{2,88e-1}$	1,134e-1 $_{2,39e-1}$	1,289e-1 $_{7,96e-2}$	5,158e-1 $_{3,47e-3}$	5,155e-1 $_{3,40e-3}$	+
ZDT4	0,000e+0 $_{0,00e+0}$	0,000e+0 $_{0,00e+0}$	0,000e+0 $_{0,00e+0}$	1,463e-1 $_{2,35e-1}$	6,551e-1 $_{5,55e-3}$	6,522e-1 $_{8,48e-3}$	+
ZDT6	3,999e-1 $_{2,22e-3}$	4,013e-1 $_{3,89e-4}$	4,013e-1 $_{6,22e-4}$	3,997e-1 $_{1,55e-2}$	4,004e-1 $_{2,01e-4}$	4,001e-1 $_{3,05e-4}$	+

Wallis con un nivel de confianza del 95 % para determinar si efectivamente las distribuciones son distintas. Los tests positivos se han marcado con un símbolo "+" en la última columna de las tablas. El mejor resultado para cada problema tiene el fondo gris. A continuación se analizan en detalle los resultados de cada experimento.

Experimento 1

Los resultados muestran que esta configuración obtiene los mejores valores de GD y HV para el problema ZDT1 aunque, por el contrario, para ZDT2 son los peores. Es más, la métrica Hypervolume ha devuelto valores nulos para ZDT2, lo que significa que todas las soluciones no dominadas calculadas están fuera de los límites del frente de Pareto real. Al aplicar esta métrica, este tipo de soluciones no se consideran puesto que, de otra manera, se habrían obtenido resultados poco fiables.

Experimento 2

En este experimento, el método simplex se sustituye por el (1+1) EA. Respecto al experimento previo, aquí AbYSS mejora los valores de GD en ZDT2, ZDT4 y ZDT6. Esta configuración también obtiene el segundo mejor valor de GD en ZDT2 y el segundo mejor valor de HV en ZDT6. No obstante, los valores de todas las métricas para ZDT4 indican que el problema no se resuelve correctamente.

Experimento 3

El Experimento 3 consiste en reducir el tamaño máximo de P, que ha pasado de 100 a 20 soluciones. Como se mencionó anteriormente, el objetivo es incrementar la intensificación realizada por AbYSS. Algunos experimentos preliminares mostraron que utilizar tamaños grandes de P tienen una mala influencia en la convergencia del algoritmo cuando se resuelven algunos MOPs. Los resultados de este experimento (Tabla I, tercera columna) reflejan que esta disminución del tamaño máximo de P permite, de forma generalizada, mejorar la métrica GD respecto a los Experimentos 1 y 2 (ZDT1 es la excepción en el Experimento 1). En concreto, los valores de GD y HV para ZDT2 son los mejores de todos los experimentos. En cuanto a la métrica de diversidad, no hay mejoras significativas en comparación con los experimentos previos.

Experimento 4

El objetivo de este experimento es investigar posibles penalizaciones en la búsqueda de AbYSS cuando modificamos el balance intensificación/diversificación a través del método de generación de subconjuntos. En este caso sólo se generan pares de individuos que pertenecen al mismo subconjunto, $RefSet_1$ o $RefSet_2$. Si bien los resultados muestran que esta configuración no alcanza el mejor valor de ninguna métrica para ninguno de los problemas considerados, sí que obtiene valores aceptables para todos ellos. El problema ZDT4 es un claro ejemplo, dónde ya no se obtiene un valor 0 para el HV. Si nos centramos en la métrica Δ, se puede observar que es los valores obtenidos aquí mejoran todos los de los experimentos previos (menos ZDT6).

Experimento 5

Manteniendo la configuración del experimento anterior, el único cambio aquí es la utilización del operador de cruce SBX en lugar de la combinación lineal de soluciones. Su uso está motivado ya que es el que se utiliza tanto en NSGA-II como en SPEA2. Los resultados muestran que esta configuración mejora sustancialmente los valores de GD para los problemas ZDT3, ZDT4 y ZDT6, mientras que mantiene valores equivalentes para los otros dos problemas. En cuanto la métrica de diversidad, esta configuración obtiene los mejores resultados para los 5 problemas resueltos. Finalmente, los valores de HV son los mejores para 3 de los 5 MOPs, mientras que las diferencias son insignificantes para los otros dos.

TABLA II

MEDIANA Y RANGO INTERCUARTÍLICO DE GD.

MOP	AbYSS $\tilde{x}_{IQR}$	NSGA-II $\tilde{x}_{IQR}$	SPEA2 $\tilde{x}_{IQR}$	
ZDT1	1,87e-4 $_{3,8e-5}$	2,19e-4 $_{4,7e-5}$	1,98e-4 $_{1,6e-5}$	+
ZDT2	1,02e-4 $_{3,8e-5}$	1,66e-4 $_{4,1e-5}$	1,26e-4 $_{3,0e-5}$	+
ZDT3	1,96e-4 $_{2,1e-5}$	2,16e-4 $_{2,1e-5}$	2,33e-4 $_{2,1e-5}$	-
ZDT4	5,40e-4 $_{3,5e-4}$	4,14e-4 $_{3,4e-4}$	5,74e-2 $_{3,3e-2}$	+
ZDT6	5,49e-4 $_{2,3e-5}$	9,91e-4 $_{9,8e-5}$	8,20e-4 $_{7,3e-5}$	+

TABLA III

MEDIANA Y RANGO INTERCUARTÍLICO DE Δ.

MOP	AbYSS $\tilde{x}_{IQR}$	NSGA-II $\tilde{x}_{IQR}$	SPEA2 $\tilde{x}_{IQR}$	
ZDT1	1,13e-1 $_{2,0e-2}$	3,86e-1 $_{6,1e-2}$	1,44e-1 $_{2,0e-2}$	+
ZDT2	1,13e-1 $_{2,0e-2}$	3,92e-1 $_{7,4e-2}$	1,49e-1 $_{3,3e-2}$	+
ZDT3	1,32e-1 $_{2,1e-1}$	3,83e-1 $_{6,8e-2}$	1,65e-1 $_{2,0e-2}$	+
ZDT4	1,24e-1 $_{3,2e-2}$	4,02e-1 $_{6,2e-2}$	4,55e-1 $_{1,7e-1}$	+
ZDT6	9,18e-2 $_{1,6e-2}$	4,35e-1 $_{4,7e-2}$	1,57e-1 $_{2,0e-2}$	+

Experimento 6

El Experimento 5 ha propuesto una configuración de AbYSS que permite resolver satisfactoriamente todos los problemas considerados. En un intento por mejorar la capacidad de búsqueda del algoritmo, en este experimento se ha incrementado el número de iteraciones del proceso de mejora, que pasa de 1 a 3 iteraciones. No obstante, los resultados que se obtienen muestran que, si bien los problemas se resuelven adecuadamente, la configuración anterior alcanza mejores valores en las métricas de rendimiento.

Se puede concluir, por tanto, que la utilización de operadores estocásticos en AbYSS hace que el algoritmo sea más robusto y preciso. También es importante mencionar, sin embargo, que hay muchas otras configuraciones alternativas con lo que aún existen posibilidades de mejora del algoritmo.

D. Comparación con NSGA-II y SPEA2

Para comprobar cómo de competitiva es nuestra propuesta, hemos comparado la mejor configuración de AbYSS respecto a NSGA-II [3] y SPEA2 [4], los dos algoritmos de referencia en el campo. Se han utilizado las versiones originales de los autores que se pueden encontrar en `www.iitk.ac.in/kangal/soft.htm` y `www.tik.ee.ethz.ch/pisa/selectors/spea2/spea2.html`, respectivamente. Ambos algoritmos utilizan el cruce SBX ($p_c = 0{,}9$) y la mutación polinomial ($p_c = 1/L$, siendo L el número de variables de decisión) como operadores genéticos. En todos los casos el número máximo de soluciones no dominadas en el frente es 100 y la condición de parada es computar 25.000 evaluaciones de funciones.

Cada algoritmo ha sido evaluado con el conjunto de problemas de prueba presentado anteriormente (Sección III-A). Los resultados para GD, Δ y HV se muestran, respectivamente, en las Tablas II, III y IV, donde se incluyen la mediana, $\tilde{x}$, y el rango intercuartílico, IQR, de 100 ejecuciones independientes. El análisis estadístico realizado es el mismo que el realizado en la sección anterior. Para cada problema, el mejor de los algoritmos tiene el fondo coloreado de gris.

TABLA IV

MEDIANA Y RANGO INTERCUARTÍLICO DE HV.

MOP	AbYSS $\tilde{x}_{IQR}$	NSGA-II $\tilde{x}_{IQR}$	SPEA2 $\tilde{x}_{IQR}$	
ZDT1	6,61e-1 $_{3,4e-4}$	6,59e-1 $_{4,4e-4}$	6,60e-1 $_{3,9e-4}$	+
ZDT2	3,28e-1 $_{3,4e-4}$	3,26e-1 $_{4,5e-4}$	3,26e-1 $_{5,0e-4}$	+
ZDT3	5,15e-1 $_{3,4e-3}$	5,14e-1 $_{1,9e-4}$	5,14e-1 $_{4,0e-4}$	+
ZDT4	6,55e-1 $_{5,5e-3}$	6,55e-1 $_{5,4e-3}$	1,06e-1 $_{2,0e-1}$	+
ZDT6	4,00e-1 $_{2,0e-4}$	3,86e-1 $_{1,6e-3}$	3,92e-1 $_{1,0e-3}$	+

Si analizamos los valores de la distancia generacional (Tabla II), se puede observar que AbYSS obtiene los mejores valores en 4 de los 5 problemas, indicando, por tanto, que los frentes a los que llega estás más próximos al frente de Pareto real que aquellos que computan NSGA-II y SPEA2. Aquí, ZDT4 es la excepción, ya que NSGA-II es el mejor para este problema.

Para la métrica de diversidad, AbYSS supera claramente los dos algoritmos de referencia en todos los problemas. Esto se traduce en que la distribución de soluciones no dominadas sobre los frentes de Pareto obtenidos por AbYSS es prácticamente perfecta. Para ilustrar esto, la Figura 1 muestra algunos frentes de Pareto alcanzados por este algoritmo.

Finalmente, los valores de la métrica HV (Tabla IV) viene a corroborar los resultados de GD y Δ. AbYSS supera a NSGA-II y SPEA2 en 4 de los 5 problemas de la familia ZDT, siendo nuevamente ZDT4 la excepción. Sin embargo, al medir tanto convergencia como diversidad, las diferencias entre AbYSS y NSGA-II para este problema son prácticamente inexistentes.

IV. CONCLUSIONES

En este trabajo se ha presentado un estudio sobre un algoritmo multiobjetivo basado en búsqueda dispersa. El algoritmo, llamado AbYSS, es una búsqueda dispersa híbrida que utiliza un archivo externo para almacenar las soluciones no dominadas que se van encontrando durante la búsqueda. Algunas de las características más importantes de AbYSS son el feedback de individuos del archivo al conjunto inicial durante la fase de reinicio, así como la combinación de dos estimadores de densidad en diferentes partes del algoritmo.

AbYSS ha sido validado utilizando una metodología estándar y actual dentro de la comunidad multiobjetivo. Se han probado diferentes configuraciones del algoritmo para evaluar su capacidad de búsqueda sobre la familia de problemas ZDT. Una vez se ha conseguido resolver satisfactoriamente estos problemas, la mejor configuración de AbYSS se ha comparado con los dos algoritmos que son el estado del arte en el campo: NSGA-II y SPEA2. Los resultados de tres métricas distintas muestran que, con el benchmark utilizado, AbYSS supera a estos dos algoritmos en convergencia (cercanía de los frentes calculados respecto al frente de Pareto óptimo) y, especialmente, en diversidad (distribución de soluciones no dominadas sobre el frente de Pareto).

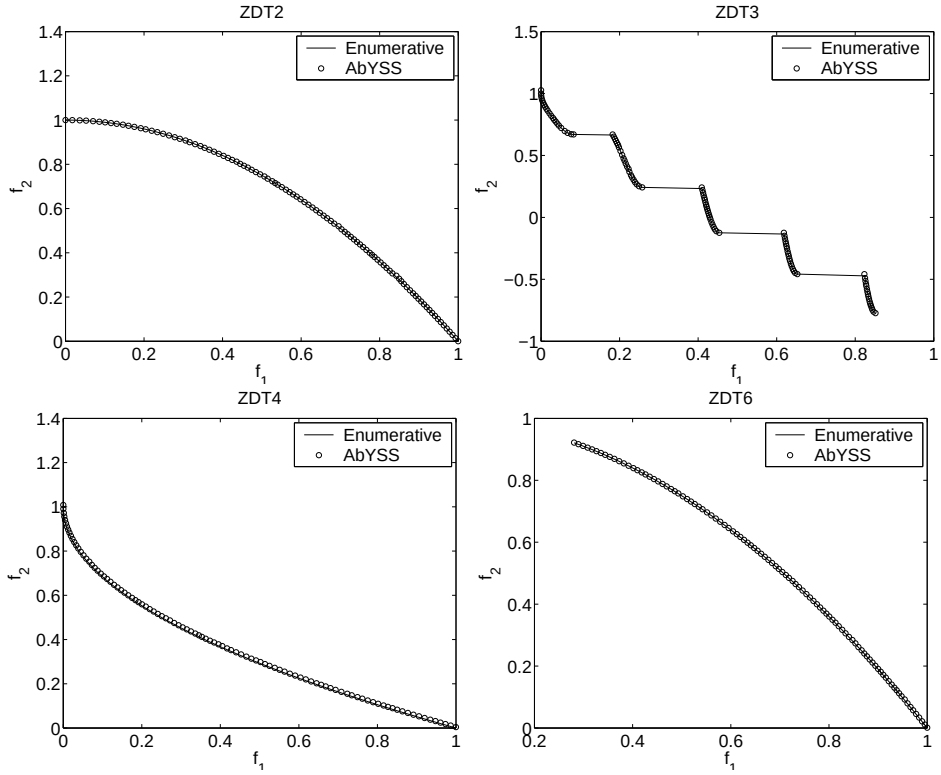

Fig. 1. Frentes de AbYSS para ZDT2, ZDT3, ZDT4 y ZDT6.

Como trabajo futuro, se pretende aplicar el algoritmo a la resolución de problemas reales. En este sentido, se intentará adaptar AbYSS para tratar con problemas de optimización combinatoria y atacar algunos problemas procedentes del campo de las telecomunicaciones.

Agradecimientos

Este trabajo ha sido realizado con el apoyo del Ministerio de Ciencia y Tecnología y el Fondo Europeo de Desarrollo Regional mediante el contrato TIN2005-08818-C04-01 (el proyecto OPLINK). Francisco Luna disfruta de una beca de la Ministerio de Ciencia y Tecnología (BOE 85/2006).

Referencias

[1] C.A. Coello, D.A. Van Veldhuizen, and G.B. Lamont, *Evolutionary Algorithms for Solving Multi-Objective Problems*, Kluwer, 2002.

[2] K. Deb, *Multi-Objective Optimization Using Evolutionary Algorithms*, John Wiley & Sons, 2001.

[3] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, 2002.

[4] E. Zitzler, M. Laumanns, and L. Thiele, "SPEA2: Improving the strength pareto evolutionary algorithm," Tech. Rep. 103, Swiss Federal Institute of Technology (ETH), Zurich, Switzerland, 2001.

[5] J. Knowles and D. Corne, "The pareto archived evolution strategy: A new baseline algorithm for multiobjective optimization," in *CEC 1999*, 1999, pp. 9–105.

[6] F. Glover, "A template for scatter search and path relinking," Number 1363 in Lecture Notes in Computer Science, pp. 13–54. Springer Verlag, 1997.

[7] F. Glover, M. Laguna, and R. Martí, "Fundamentals of scatter search and path relinking," *Control and Cybernetics*, vol. 29, no. 3, pp. 653–684, 2000.

[8] F. Glover, M. Laguna, and R. Martí, "Scatter search," in *Advances in Evolutionary Computing: Theory and Applications*, pp. 519–537. Springer, 2003.

[9] R. P. Beausoleil, "MOSS: Multiobjective scatter search applied to nonlinear multiple criteria optimization," *European Journal of Operational Research*, vol. 169, no. 2, pp. 426–449, March 2006.

[10] J. Molina, M. Laguna, R. Martí, and R. Caballero, "SSPMO: A scatter tabu search procedure for non-linear multiobjective optimization," To be published in INFORMS Journal on Computing, 2005.

[11] A. J. Nebro, F. Luna, and E. Alba, "New ideas in applying scatter search to multiobjective optimization," in *EMO 2005*, C.A. Coello, A. Hernández, and E. Zitler, Eds., 2005, vol. 3410 of *LNCS*, pp. 443–458.

[12] A.J. Nebro, F. Luna, E. Alba, A. Beham, and B. Dorronsoro, "AbYSS: Adapting Scatter Search for Multiobjective Optimization," Tech. Rep. ITI-2006-2, Depto. de Lenguajes y Ciencias de la Computación, 2006.

[13] E. Zitzler, K. Deb, and L. Thiele, "Comparison of multiobjective evolutionary algorithms: Empirical results," *Evolutionary Computation*, vol. 8, no. 2, pp. 173–195, 2000.

[14] K. Deb and B. Agrawal, "Simulated Binary Crossover for Continuous Search Space," *Complex Systems*, vol. 9, no. 2, pp. 115–148, 1995.

[15] C.A. Coello, G. Toscano, and M. Salazar, "Handling multiple objectives with particle swarm optimization," *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 3, pp. 256–278, June 2004.

[16] D. A. Van Veldhuizen and G. B. Lamont, "Multiobjective Evolutionary Algorithm Research: A History and Analysis," Tech. Rep. TR-98-03, Dept. Elec. Comput. Eng., 1998.

[17] A. Zhou, Y. Jin, Q. Zhang, B. Sendhoff, and E. Tsang, "Combining model-based and genetics-based offspring generation for multi-objective optimization using a convergence criterion," in *CEC 2006*, 2006, pp. 3234–3241.

[18] J.A. Nelder and R. Mead, "A simplex method for function minimization," *Computer Journal*, vol. 7, pp. 308–313, 1965.

[19] F. Herrera, M. Lozano, and D. Molina, "Continuous scatter search: An analysis of the integration of some combination methods and improvement strategies," *European Journal of Operational Research*, vol. 169, pp. 450–476, 2006.