Un Nuevo Algoritmo Evolutivo en Programación Multiobjetivo para Aproximar el Frente Óptimo de Pareto

Mariano Luque, Ana B. Ruiz y Rubén Saborido

Resumen— **En este trabajo, presentamos un nuevo algoritmo evolutivo para Programación Multiobjetivo que intenta aproximar todo el frente óptimo de Pareto. En dicho algoritmo, consideramos como función fitness una función escalarizada de logro, que es una extensión de la distancia Tchebychev o minimax. Paralelamente, tomamos dos puntos de referencia, el utopía y el nadir, para clasificar los individuos de cada generación en distintas fronteras. Para asegurar la diversidad de soluciones, se emplea un conjunto de vectores de pesos en la función de logro, que verifican que los vectores formados por las componentes inversas están distribuidos uniformemente. Finalmente, se muestra el funcionamiento del algoritmo propuesto a través de un estudio comparativo con MOEA/D y NSGA-II para varios problemas con tres y cinco funciones objetivo, respecto a la métrica hipervolumen.**

Palabras clave— **Optimización multiobjetivo, Algoritmo evolutivo, Pareto optimalidad, Función escalarizada de logro, Punto de referencia.**

I. Introducción

Numerosos problemas reales de decisión requieren optimizar varios criterios al mismo tiempo que, por lo general, suelen estar en conflicto. A través de la modelización matemática de estos criterios, conseguimos formular un problema de optimización multiobjetivo, que consiste en maximizar o minimizar dichos criterios, denominados funciones objetivo, bajo ciertas restricciones. Por lo general, es imposible encontrar una única solución que optimice todas las funciones objetivo al mismo tiempo. Sin embargo, existe un conjunto de *soluciones óptimas de Pareto* en las que no es posible mejorar una función objetivo sin empeorar alguna otra. El conjunto de todas las soluciones óptimas de Pareto en el espacio de decisión se denomina *conjunto óptimo de Pareto* y su imagen en el espacio de objetivos es el *frente óptimo de Pareto*.

Desde los años ochenta, la *Optimización Multiobjetivo Evolutiva* [Alba et al., 2009], [Coello et al., 2007], [Deb, 2001] (*Evolutionary Multiobjective Optimization (EMO)* en inglés) ha tratado de resolver los problemas de optimización multiobjetivo. Estas técnicas trabajan con una población de individuos (soluciones) y evolucionan de una población a otra a través de operadores como mutación, recombinación y selección. El principal propósito de los algoritmos EMO es aproximar todo el frente óptimo de Pareto a través de un conjunto de soluciones no dominadas entre sí. Para que dichas soluciones representen adecuadamente el frente óptimo de Pareto, deben estar próximas al frente y estar distribuidas uniformemente. Pero conseguir convergencia y diversidad al mismo tiempo no es tarea fácil, especialmente cuando aumenta el número de objetivos. Algoritmos evolutivos como NSGA-II [Deb et al., 2002a] y MOEA/D [Zhang and Li, 2007], entre otros, han sido aplicados y se siguen aplicando con éxito a numerosos problemas de optimización multiobjetivo y aplicaciones reales.

En este trabajo, proponemos un nuevo algoritmo evolutivo, que denominamos *Global WASF-GA (GWASF-GA)*, que extiende las ideas básicas del algoritmo evolutivo basado en preferencias propuesto en [Ruiz et al., 2014]. En el nuevo algoritmo, se aproxima el frente óptimo de Pareto considerando la función escalarizada de logro sugerida en [Wierzbicki, 1980] como función fitness. Una función de logro es una función real que combina las funciones objetivo originales con un punto de referencia formado por niveles de aspiración para cada objetivo. Minimizar la función de logro propuesta en [Wierzbicki, 1980] significa, en la práctica, proyectar el punto de referencia sobre el frente óptimo de Pareto en la dirección determinada por los inversos de los pesos considerados.

En dicha función de logro, nuestro algoritmo emplea simultáneamente los vectores utopía y nadir como puntos de referencia, y considera un conjunto de vectores de pesos que determinan internamente un conjunto de direcciones de búsqueda de las nuevas soluciones. En base al significado práctico de los pesos en la función de logro, GWASF-GA considera vectores de pesos cuyos vectores formados por sus componentes inversas están uniformemente distribuidos. De esta forma, mediante el uso simultáneo del utopía y el nadir y los vectores de pesos indicados, la población final puede adaptarse mejor a la topología del frente óptimo de Pareto, alcanzando una buena distribución de soluciones.

A continuación, en la Sección II introducimos los conceptos necesarios de Programación Multiobjetivo. El nuevo algoritmo GWASF-GA se describe en

Departamento de Economía Aplicada (Matemáticas), Universidad de Málaga (España). E-mail: mluque@uma.es
Departamento de Economía Aplicada (Matemáticas), Universidad de Málaga (España). E-mail: abruiz@uma.es
Polytechnique Montréal Researchers in Software Engineering, École Polytechnique de Montréal (Canada). E-mail: ruben.saborido-infantes@polymtl.ca

la Sección III, señalando la contribución de nuestro método respecto a los ya existentes. En la Sección IV, se muestran varios experimentos computacionales, en los que se compara GWASF-GA con los algoritmos MOEA/D y NSGA-II. Finalmente, concluimos en la Sección V con las conclusiones.

II. Programación Multiobjectivo

Un *problema de optimización multiobjetivo* viene dado por la siguiente formulación:

$$\begin{array}{ll} \text{minimizar} & \{f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_k(\mathbf{x})\} \\ \text{sujeto a} & \mathbf{x} \in \mathbf{S}, \end{array} \quad (1)$$

donde se optimizan $k \geq 2$ funciones objetivo $f_i : S \rightarrow \mathbb{R}$ en conflicto. Los *vectores de decisión* $\mathbf{x} = (x_1, x_2, \ldots, x_n)^T$ pertenecen al llamado conjunto factible $S \subset \mathbb{R}^n$, el cual puede ser continuo o no dependiendo del tipo de variables (continuas o discretas) y/o el tipo de restricciones. La imagen $\mathbf{z} = \mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_k(\mathbf{x}))^T$ de cualquier $\mathbf{x} \in S$ contiene los valores en las funciones objetivo y se denomina *vector criterio*. El conjunto de todos los vectores criterio $Z = \mathbf{f}(S) \subset \mathbb{R}^k$ constituye el espacio de objetivos.

En la mayoría de los casos, el conflicto existente entre los objetivos imposibilita encontrar una solución en la que todos los objetivos alcancen sus óptimos individuales simultáneamente. Sin embargo, hay un conjunto de soluciones compromiso: las denominadas *soluciones eficientes* o *Pareto óptimas*. Para un problema multiobjetivo como (1), dados dos vectores $\mathbf{z}, \mathbf{z}' \in \mathbb{R}^k$, decimos que $\mathbf{z}$ *domina a* $\mathbf{z}'$ si y solo si $z_i \leq z'_i$ para todo $i = 1, \ldots, k$ y existe algún j tal que $z_j < z'_j$. Se considera que $\mathbf{x} \in S$ es una solución *Pareto óptima* del problema (1) si no existe ningún $\mathbf{x}' \in S$ tal que $\mathbf{f}(\mathbf{x}')$ domine a $\mathbf{f}(\mathbf{x})$. El conjunto de todas las soluciones Pareto óptimas se denomina *conjunto eficiente* o *conjunto óptimo de Pareto*, denotado por E. Su imagen en el espacio de objetivos se conoce como *frontera eficiente* o *frente óptimo de Pareto*, denotado por $\mathbf{f}(E)$.

Otros conceptos importantes en programación multiobjetivo son los vectores ideal y nadir. Respectivamente, sus componentes proporcionan cotas inferiores y superiores para las funciones objetivo y representan los mejores y peores valores que cada función puede lograr en el frente óptimo de Pareto. El *vector ideal*, $\mathbf{z}^\star = (z_1^\star, \ldots, z_k^\star)^T \in \mathbb{R}^k$, se calcula minimizando cada función objetivo individualmente en el conjunto factible, esto es, $z_i^\star = \min_{\mathbf{x} \in S} f_i(\mathbf{x}) = \min_{\mathbf{x} \in E} f_i(\mathbf{x})$ para todo $i = 1, \ldots, k$. El *vector nadir*, $\mathbf{z}^{\text{nad}} = (z_1^{\text{nad}}, \ldots, z_k^{\text{nad}})^T$, se obtiene maximizando cada función en el conjunto eficiente, $z_i^{\text{nad}} = \max_{\mathbf{x} \in E} f_i(\mathbf{x})$ para todo $i = 1, \ldots, k$. En la práctica, el vector nadir es difícil de calcular y por ello se suele estimar [Deb and Miettinen, 2010]. A veces, es útil trabajar con un vector que domina a cualquier solución óptima de Pareto, lo cual se puede conseguir construyendo un vector que domina estrictamente al vector ideal. Este vector se denomina *vector utopía*, $\mathbf{z}^{\star\star} = (z_1^{\star\star}, \ldots, z_k^{\star\star})^T$, y se define como $z_i^{\star\star} = z_i^\star - \varepsilon_i$ para todo $i = 1, \ldots, k$, donde $\varepsilon_i > 0$ es una cantidad real pequeña.

Como señalábamos anteriormente, el nuevo algoritmo propuesto en este trabajo se basa en el uso de la función escalarizada de logro propuesta en [Wierzbicki, 1980]. Para un punto de referencia $\mathbf{q} = (q_1, \ldots, q_k)^T$, formado por valores de aspiración q_i deseables para cada función objetivo, y un vector de pesos $\mu = (\mu_1, \ldots, \mu_k)^T$, con $\mu_i > 0$ para todo $i = 1, \ldots, k$, esta función viene dada por:

$$\begin{array}{rcl} s(\mathbf{q}, \mathbf{f}(\mathbf{x}), \mu) & = & \max\limits_{i=1,\ldots,k} \{ \mu_i (f_i(\mathbf{x}) - q_i) \} + \\ & & \rho \sum_{i=1}^{k} \mu_i (f_i(\mathbf{x}) - q_i). \end{array} \quad (2)$$

donde $\rho > 0$ un valor real suficientemente pequeño (normalmente $\rho = 0{,}001$). Está probado que la solución factible que minimiza (2) es una solución eficiente del problema original (1) [Miettinen, 1999]. Conviene tener en cuenta que si los rangos de las funciones objetivo están en escalas muy diferentes, los valores de las funciones objetivo deben normalizarse en (2), por ejemplo, dividiendo $f_i(\mathbf{x}) - q_i$ por $z_i^{\text{nad}} - z_i^\star$, para todo $i = 1, \ldots, k$.

En la práctica, si $\mathbf{q}$ es alcanzable (esto es, si existe alguna solución Pareto óptima en la que se alcanzan o se mejoran todos los valores de aspiración de $\mathbf{q}$), la solución que minimiza (2) mejora los valores de aspiración tanto como sea posible. En caso contrario, si $\mathbf{q}$ es inalcanzable, minimizar (2) equivale a la distancia minmax. Además, si los pesos considerados son estrictamente positivos, minimizar (2) significa proyectar $\mathbf{q}$ sobre el frente óptimo de Pareto en la dirección determinada por los inversos de los pesos.

Está demostrado que se puede obtener cualquier solución óptima de Pareto minimizando (2) sobre S, usando el vector ideal $\mathbf{z}^\star$ como punto de referencia (o cualquier vector que lo domine), y variando el vector de pesos [Miettinen, 1999], [Wierzbicki, 1986]. Esto permite generar todo tipo de frentes óptimos de Pareto, incluidos aquellos con zonas convexas ya que es capaz de generar las llamadas soluciones no soportadas [Steuer, 1986]. Todo esto también es válido si consideramos el vector nadir $\mathbf{z}^{\text{nad}}$ como punto de referencia (o cualquier vector dominado por él) [Miettinen, 1999].

III. Global WASF-GA

El nuevo algoritmo *Global WASF-GA (GWASF-GA)* que describimos a continuación puede considerarse como una extensión del algoritmo evolutivo basado en preferencias WASF-GA [Ruiz et al., 2014] para aproximar todo el frente óptimo de Pareto. En términos generales, en cada generación de GWASF-GA, se hace una clasificación de los individuos en fronteras en base a los valores que alcanzan en la función escalarizada de logro (2), considerando simultáneamente dos puntos de referencia: un vector utopía y el vector nadir. En dicha clasificación, se prefieren las soluciones con lo valores más pequeños

de (2) para cada uno de los dos puntos de referencia. Por otro lado, la diversidad de soluciones se asegura a través de un conjunto de vectores de pesos que se genera inicialmente. Dichos vectores de pesos se emplean en la clasificación de los individuos a través de la función de logro, de tal forma que cada vector define una dirección de búsqueda de nuevas soluciones próximas al frente óptimo de Pareto.

Como ya hemos indicado, si consideramos un punto de referencia que domina el vector ideal, por ejemplo un vector utopía, se puede generar cualquier solución óptima de Pareto minimizando (2) y variando el vector de pesos [Miettinen, 1999], [Wierzbicki, 1986]. El mismo resultado es válido si el punto de referencia es el vector nadir o cualquier vector dominado por él [Miettinen, 1999]. De acuerdo con esto, GWASF-GA aproxima el frente óptimo de Pareto de la siguiente forma: en cada generación, se minimiza la función de logro (2), tomando al mismo tiempo el utopía y el nadir como puntos de referencia, y considerando un conjunto de vectores de pesos. Consecuentemente, cualquier solución óptima de Pareto puede ser generada por nuestro algoritmo en base a los resultados anteriores.

Para asegurar la diversidad de soluciones, se tiene en cuenta que la función (2) proyecta el punto de referencia sobre el frente óptimo de Pareto en la dirección dada por los inversos de los pesos usados, como ya se señaló en la Sección II. Debido a ello, es importante que los vectores de pesos considerados en GWASF-GA definan un conjunto de direcciones de proyección distribuidas uniformemente. La idea es que, si las direcciones de proyección están distribuidas de manera uniforme, es bastante probable que las soluciones no dominadas obtenidas estén también distribuidas por todo el frente óptimo de Pareto en la mayoría de los problemas. En problemas con frentes óptimos de Pareto discontinuos es donde puede ser que la distribución de soluciones no sea tan buena. Esto se debe a que en esos casos, minimizar la función (2) en el conjunto factible S nos puede dar la misma solución para vectores de pesos distintos (véase [Luque et al., 2007], [Luque et al., 2015]).

Respecto a la manera de generar los pesos, sea N_μ el número de vectores de pesos que queremos generar (en las pruebas computacionales haremos coincidir con el número de pesos del MOEA/D). Sea $\{\mathbf{u}^1, \mathbf{u}^2, \ldots, \mathbf{u}^{N_\mu}\}$ un conjunto de vectores de pesos equidistribuidos en el espacio $(0,1)^k$, los cuales pueden ser generados de varias maneras [Scheffe, 1958], [Scheffe, 1963], [Wagner et al., 2013]. El conjunto de vectores de pesos $\{\mu^1, \ldots, \mu^{N_\mu}\}$ que vamos a considerar en Global WAS-GA son los inversos de las componentes de los $\mathbf{u}^j = (u_1^j, \ldots, u_k^j)$, es decir, para todo $j = 1, \ldots, N_\mu$, el vector de pesos $\mu^j = (\mu_1^j, \ldots, \mu_k^j)$ considerado en la función (2) viene definido por:

$$\mu_i^j = \frac{1}{u_i^j} \quad \text{for every } i = 1, \ldots, k. \tag{3}$$

La manera de generar $\{\mathbf{u}^1, \mathbf{u}^2, \ldots, \mathbf{u}^{N_\mu}\}$ en nuestro caso ha sido a través de un método uniforme sobre el conjunto $[\varepsilon, 1-\varepsilon]^k$ con $\varepsilon > 0$ una cantidad pequeña (por ejemplo $\varepsilon = 0{,}01$), evitando así componentes nulas. Este conjunto de vectores ha sido filtrado por el método de las k-medias [MacQueen, 1967] para obtener el número de vectores buscado. Para más detalles, véase [Ruiz et al., 2014].

La Figura 1 ofrece una idea gráfica del funcionamiento de GWASF-GA para un problema de optimización biobjetivo, donde hemos representado el vector nadir y un vector utopía. Las flechas representan las direcciones de búsqueda definidas por un conjunto de vectores de pesos dado. Desde el punto de vista práctico, se puede observar cómo el frente óptimo de Pareto se aproxima mediante la proyección simultánea del utopía y el nadir, teniendo en cuenta las múltiples direcciones de búsqueda. Debe decirse que en nuestro algoritmo consideramos el vector utopía en lugar del ideal porque puede resultar más fácil aproximar los extremos de el frente óptimo de Pareto en este caso [Miettinen, 1999]. Asimismo, en lugar del vector nadir, emplearemos un vector dominado por él. Para obtener dicho vector, se han empeorado ligeramente las componentes del nadir. Sin embargo, para simplificar la notación, hemos seguido denotando dicho vector por $\mathbf{z}^{\text{nad}}$.

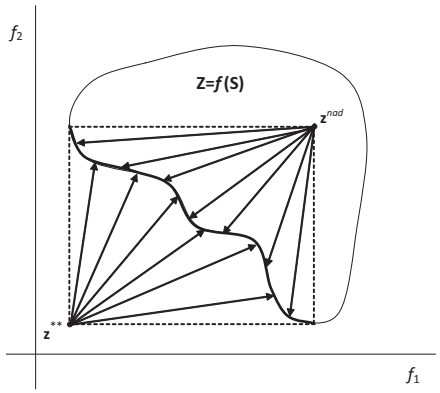

Fig. 1
Idea gráfica del GWASF-GA para un problema de optimzación biobjetivo.

Sea N la dimensión de la población, h la actual generación, P^h la población de individuos en la generación h, Q^h la población de hijos en la generación h y F_n^h la n-ésima frontera en la generación h. Denotaremos por $\#(A)$ el numero de elementos del conjunto A. El Algoritmo 1 describe los principales pasos de GWASF-GA.

Inicialmente, los vectores ideal y nadir se aproximan considerando, respectivamente, los mejores y peores valores que alcanzan las funciones objetivo en la población inicial creada aleatoriamente. Posteriormente, se obtienen el utopía y la ligera modificación del nadir (por ejemplo, sumando a cada componente la cantidad ε_i que se considera para obtener el

utopía). El hecho de considerar ambos puntos de referencia hace que si alguno de los dos no esté bien estimado, sea compensando con el otro a la hora de obtener una buena aproximación del frente óptimo de Pareto. Es poco probable que se consideren malas estimaciones de ambos vectores al mismo tiempo y, en dicho caso, suele corregirse a medida que avanzan las iteraciones.

Como puede observarse, en cada generación h, después de generar la población de hijos Q^h a través de los operadores de selección, recombinación y mutación, disponemos de una población total $P = P^h \cup Q^h$ de dimensión $2N$ (padres e hijos). Esta población P se divide a continuación en varias fronteras según los valores que alcanzan los individuos en la función de logro, clasificando inicialmente las soluciones factibles y luego las infactibles. La primera frontera está formada, por un lado, por las soluciones factibles de P que alcanzan los menores valores de (2) para $\mathbf{z}^{\star\star}$ considerando la mitad de los vectores de pesos (los impares) y, por otro lado, por las soluciones factibles de P que consiguen los menores valores de (2) para $\mathbf{z}^{\text{nad}}$ considerando la otra mitad de los vectores de pesos (los pares). Estas soluciones se eliminan de P y, posteriormente, se forma una nueva frontera siguiendo el mismo procedimiento. Así se seguiría sucesivamente hasta acabar con las soluciones factibles de P. Luego, se procede a clasificar las soluciones infactibles en fronteras teniendo en cuenta la violación global de las restricciones, según el procedimiento descrito en [Deb et al., 2002a].

Finalmente, se incluyen en la población P^{h+1} de la siguiente generación los N individuos de las primeras fronteras. En caso de que la última frontera considerada contenga más soluciones que el número de soluciones que se pueden incluir en P^{h+1} para tener N individuos, se incluyen aquellas soluciones de dicha frontera con los menores valores de (2).

Téngase en cuenta que puede ocurrir que una misma solución logre el mínimo valor de (2) para más de un vector de pesos. En ese caso, para lograr una mayor diversidad, no se permite repetir dicha solución en la misma frontera y se tomaría la siguiente solución con el menor valor de (2).

Conviene decir que, en base a nuestros experimentos, se consiguen mejores resultados cuando el utopía y el nadir se consideran al mismo tiempo, que cuando sólo se emplea uno de ellos. Usar ambos puntos permite al proceso de búsqueda adaptarse mejor a la forma de el frente óptimo de Pareto.

A continuación, señalamos las diferencias de GWASF-GA con respecto a algunos algoritmos evolutivos con los que comparte ciertos aspectos:
1. El algoritmo WASF-GA, en el que está basado GWASF-GA, concentra la búsqueda de soluciones en una región concreta del frente óptimo de Pareto (la región de interés definida por las preferencias del decisor). A diferencia, GWASF-GA está diseñado para aproximar el frente óptimo de Pareto en su totalidad. Además, GWASF-GA considera dos puntos

Algoritmo 1 Algoritmo GWASF-GA

Entrada: N_μ vectores de pesos (μ^j para $j = 1, \ldots, N_\mu$) y el *criterio de parada*.
Salida: Un conjunto P_{final} que aproxima el frente óptimo de Pareto.

1: $h = 0$.
2: Generar de forma aleatoria una población inicial P^0 con N individuos.
3: **para** $i = 1, \ldots, k$ **hacer**
4: $z_i^\star = \min_{\mathbf{x} \in P^0} f_i(\mathbf{x})$ y $z_i^{\text{nad}} = \max_{\mathbf{x} \in P^0} f_i(\mathbf{x})$.
5: $z_i^{\star\star} = z_i^\star - \varepsilon_i$ y $z_i^{\text{nad}} = z_i^{\text{nad}} + \varepsilon_i$.
6: **fin para**
7: **mientras** *criterio de parada = falso* **hacer**
8: $P = \emptyset$.
9: Aplicar los operadores de selección, recombinación y mutación a la población P^h para generar una población Q^h con N hijos. Sea $P = P^h \cup Q^h$.
10: Si existe $\mathbf{x} \in Q^h$ tal que $f_i(\mathbf{x}) < z_i^\star$ para algún $i = 1, \ldots, k$, actualizar $z_i^\star = f_i(\mathbf{x})$. Posteriormente, actualizar $z_i^{\star\star} = z_i^\star - \varepsilon_i$.
11: Si $h > 0$ y existe $\mathbf{x} \in Q^h$ tal que $f_i(\mathbf{x}) > z_i^{\text{nad}}$ para algún $i = 1, \ldots, k$, actualizar $z_i^{\text{nad}} = f_i(\mathbf{x}) + \varepsilon_i$.
12: Para cada $\mathbf{x} \in P$ y cada $j = 1, \ldots, \frac{N_\mu}{2}$, calcular $s(\mathbf{z}^{\star\star}, \mathbf{f}(\mathbf{x}), \mu^{2j-1})$ y $s(\mathbf{z}^{\text{nad}}, \mathbf{f}(\mathbf{x}), \mu^{2j})$.
13: $n = 0$.
14: **mientras** $P \neq \emptyset$ y existe alguna solución factible en P **hacer**
15: $n = n + 1$.
16: $F_n^h = \emptyset$.
17: **para** $j = 1, \ldots, \frac{N_\mu}{2}$ **hacer**
18: Si $\mathbf{x} \in P$ la solución factible con el menor valor de $s(\mathbf{z}^{\star\star}, \mathbf{f}(\mathbf{x}), \mu^{2j-1})$, entonces $F_n^h = F_n^h \cup \{\mathbf{x}\}$ y eliminar $\mathbf{x}$ de P.
19: Si $\mathbf{x} \in P$ la solución factible con el menor valor de $s(\mathbf{z}^{\text{nad}}, \mathbf{f}(\mathbf{x}), \mu^{2j})$, entonces $F_n^h = F_n^h \cup \{\mathbf{x}\}$ y eliminar $\mathbf{x}$ de P.
20: **fin para**
21: **fin mientras**
22: **mientras** $P \neq \emptyset$ y existe alguna solución infactible en P **hacer**
23: Clasificar las soluciones infactibles de P en base a la violación global de las restricciones [Deb et al., 2002a].
24: **fin mientras**
25: $P^{h+1} = \emptyset$.
26: $m = 1$.
27: **mientras** $\#(P^{h+1} \cup F_m^h) \leq N$ **hacer**
28: $P^{h+1} = P^{h+1} \cup F_m^h$.
29: $m = m + 1$.
30: **fin mientras**
31: Para el valor m con $\#(P^{h+1} \cup F_m^h) > N$, introducir en P^{h+1} los individuos de F_n^h con los valores más pequeños de (2) hasta que $\#(P^{h+1}) = N$.
32: $h = h + 1$.
33: **fin mientras**
34: $P_{final} = F_1^{h-1}$.

de referencia simultáneamente, el utopía y el nadir (que no contienen preferencias), mientras WASF-GA emplea únicamente uno que proporciona el decisor.
2. GWASF-GA y NSGA-II comparten la idea de clasificar los individuos en varias fronteras. Sin embargo, GWASF-GA lo hace en función de los valores que alcanzan los individuos para una función escalarizada de logro y NSGA-II clasifica los individuos en base a la relación de dominancia existente entre ellos. Es conocido que los algoritmos evolutivos basados en dominancia tienen problemas de convergencia en presencia de un número elevado de funciones objetivo [Ishibuchi et al., 2008a], [Ishibuchi et al., 2008b]. Estas dificultades para converger se pueden aliviar en cierta medida si se emplean funciones escalarizadas como funciones fitness, porque permiten la escalabilidad del problema multiobjetivo a un problema mono-objetivo, sea cual sea el número de objetivos. Por otro lado, para mantener la diversidad, GWASF-GA emplea un conjunto de direcciones de búsqueda uniformemente distribuidas (las que definen los vectores de pesos) y NSGA-II hace uso de una distancia 'crowding'.
3. Por último, GWASF-GA y MOEA/D también tienen ciertas similitudes. Ambos algoritmos usan funciones escalarizadas como funciones fitness y, además, emplean varios vectores de pesos para generar nuevas soluciones. Sin embargo, el criterio para seleccionar las mejores soluciones de cada generación es totalmente diferente. MOEA/D minimiza la función escalarizada para cada vector de pesos y el vector ideal, basándose en la idea de que vectores de pesos cercanos poseen soluciones cercanas. En cambio, GWASF-GA clasifica los individuos por fronteras y selecciona aquéllos que están en las primeras fronteras, que son los que alcanzan menores valores de la función escalarizada de logro para el utopía y el nadir y para cada vector de pesos. Otro aspecto a considerar son las funciones de logro consideradas en GWASF-GA (la función (2)) y en MOEA/D (la distancia de Tchebychev), aunque esto realmente no se puede considerar como una diferencia intrínseca entre ambos algoritmos ya que, en ambos casos, podríamos emplear otras funciones. Sin embargo, conviene tener en cuenta la diferencia entre ambas funciones, que está en el término aumentado y sirve para asegurar la eficiencia de la solución obtenida. En muchos problemas, donde no hay soluciones debílmente eficientes[1] que no sean eficientes, lo más normal es que las dos funciones produzcan resultados idénticos. Sin embargo, en problemas con soluciones débilmente eficientes que no sean eficientes, los resultados sí que pueden diferir ya que la distancia de Tchebychev sólo permite asegurar la eficiencia débil de la solución (para más detalle, véase [Miettinen, 1999].

[1]Una solución $\mathbf{x} \in S$ es *débilmente eficiente o débilmente Pareto óptima* si no existe ningún $\mathbf{x}' \in S$ tal que $\mathbf{f}(\mathbf{x}')$ domine estrictamente a $\mathbf{f}(\mathbf{x})$, esto es, no existe $\mathbf{x}' \in S$ tal que $f_i(\mathbf{x}') < f_i(\mathbf{x})$ para todo $i = 1, \ldots, k$.

IV. Experimientos computacionales

Se han llevado a cabo una serie de experimentos computacionales en los que se ha comparado las poblaciones obtenidas por GWASF-GA, MOEA/D y NSGA-II[2]. Se han considerado problemas test con tres y cinco funciones objetivo de las familias DTLZ [Deb et al., 2002b], WFG [Huband et al., 2007], LZ09 [Li and Zhang, 2009] y CEC09 [Zhang et al., 2008]. La calidad de las poblaciones finales se ha comparado en función de la métrica hipervolumen (HV) [Zitzler and Thiele, 1999] obtenida en 30 ejecuciones independientes. Para una población P, $HV(P)$ es el hipervolumen del espacio de objetivos que está dominado por las soluciones de P y está acotado por un punto de referencia $\mathbf{r}$ dominado por todas las soluciones de P. En cada problema, se ha considerado que $\mathbf{r}$ es el vector nadir $\mathbf{z}^{\text{nad}}$. Cuanto más grande sea el hipervolumen, mejor será la aproximación del frente óptimo de Pareto.

A. Parámetros empleados

- Se han realizado 30 ejecuciones independientes de cada algoritmo para cada problema.
- Para los problemas con tres objetivos, hemos considerado 300 individuos y se han ejecutado 400 generaciones. En el caso de cinco objetivos, hemos usado 1000 individuos y 600 generaciones.
- En GWASF-GA y NSGA-II, el operador de recombinación ha sido el SBX [Deb, 2001], con $\eta_c = 20$ y $P_c = 0{,}9$. En MOEA/D, se ha considerado el operador diferencial [Li and Zhang, 2009] como operador de recombinación, con $CR = 1{,}0$ and $F = 0{,}5$.
- El operador de mutación polinomial [Deb, 2001] se ha empleado en los tres algoritmos, con $\eta_m = 20$ y $P_m = 1/n$, donde n es el número de variables.
- El *criterio de parada* considerado en todos los algoritmos ha sido alcanzar el número máximo de generaciones.
- En GWASF-GA y MOEA/D, hemos considerado tantos vectores de pesos como tamaño de población, esto es, $N_\mu = 300$ para los problemas con tres objetivos y $N_\mu = 1000$ para los de cinco objetivos. En GWASF-GA, los vectores de pesos han sido generados siguiendo el procedimiento descrito en [Ruiz et al., 2014], y en MOEA/D se han obtenido de acuerdo con [Zhang and Li, 2007].
- Para obtener el vector utopía $\mathbf{z}^{\star\star}$ y la ligera modificación del nadir en GWASF-GA, hemos tomado ε_i igual al 0,1 % de la diferencia entre los correspondientes valores del nadir y del ideal, para cada $i = 1, \ldots, k$.
- En MOEA/D, hemos empleado los siguientes valores para los parámetros de control: $T = 20$, $\delta = 0{,}9$ y $n_r = 2$.

[2]GWASF-GA se ha implementado en Java y se ha incorporado a la plataforma jMetal [Durillo et al., 2010], [Durillo and Nebro, 2011], que proporciona implementaciones de MOEA/D y NSGA-II.

En nuestros experimentos, los vectores de pesos considerados en GWASF-GA se han generado siguiendo el procedimiento descrito en [Ruiz et al., 2014] porque, tras realizar varias pruebas comparando los resultados obtenidos al usar otros procedimientos, éste fue el que alcanzó mejores resultados.

B. Resultados

La Tabla I muestra el valor medio y la desviación típica alcanzados por el HV en 30 ejecuciones de cada algoritmo. Para cada problema, se ha resaltado en negrita el algoritmo que ha obtenido el mejor valor medio. De acuerdo a esta tabla, para los problemas con tres objetivos, GWASF-GA alcanza el mejor valor medio del HV en 14 problemas, mientras que MOEA/D y NSGA-II solo consiguen mejor HV medio que el resto de algoritmos en tres casos cada uno. En los problemas con cinco objetivos, GWASF-GA obtiene el mejor valor medio del HV en ocho problemas, MOEA/D en un problema y NSGA-II no gana en ningún caso.

Para validar los resultados presentados, hemos realizado el test estadístico Wilcoxon [Wilcoxon, 1945]. Este test estadístico es una prueba no paramétrica que compara dos muestras y determina si existen diferencias entre ellas. En nuestro caso, este test se ha llevado a cabo para determinar si la diferencia entre los valores medios de HV alcanzados por cada par de algoritmos es estadísticamente significativa o no. La Tabla II muestra los resultados obtenidos para los problemas con tres y cinco funciones objetivo. Se ha usado un nivel significativo del 5 % y la hipótesis nula es *'los dos algoritmos alcanzan el mismo valor medio del HV'*. Si p-valor $< 0{,}05$, la hipótesis nula se rechaza y entonces los valores medios del HV alcanzados por los dos algoritmos considerados son comparables. En este caso, en la Tabla II, el símbolo ▲ indica que el algoritmo de la fila (GWASF-GA) alcanza el mejor valor medio del HV, mientras que el símbolo ▽ aparece si el algoritmo de la columna (MOEA/D o NSGA-II, respectivamente) obtiene el mejor valor medio del HV. Si la hipótesis nula es cierta (p-valor $\geq 0{,}05$), se emplea el símbolo – para indicar que la comparativa no es estadísticamente significativa.

De acuerdo con la Tabla II, se puede observar que los datos son estadísticamente significativos en la mayoría de los casos. En cuanto a los problemas con tres objetivos, los resultados indican que, de los 20 problemas considerados, GWASF-GA obtiene mejor HV medio que MOEA/D en 16 problemas, MOEA/D gana a GWASF-GA únicamente en dos problemas y la diferencia entre ambos datos no es significativa en dos casos. En comparación con NSGA-II, GWASF-GA alcanza mejor HV medio en 16 problemas, NSGA-II sólo gana en dos casos y dichos valores no son significativamente diferentes en otros dos problemas. En relación a los nueve problemas con cinco objetivos, GWASF-GA alcanza mejores valores medios del HV que MOEA/D en ocho casos, mientras que MOEA/D sólo le gana en un problema. El mismo análisis se observa al comparar GWASF-GA con NSGA-II.

Teniendo en cuenta que la métrica hipervolumen mide tanto la diversidad como la convergencia de las poblaciones finales, los resultados obtenidos en este estudio sugieren que GWASF-GA aproxima el frente óptimo de Pareto alcanzando mejores niveles de convergencia y diversidad que MOEA/D y NSGA-II, en los problemas con tres y cinco objetivos considerados.

V. Conclusiones

En este trabajo, se ha propuesto un nuevo algoritmo evolutivo de optimización multiobjetivo denominado Global WASF-GA (GWASF-GA). Teniendo en cuenta un conjunto de vectores de pesos, un vector utopía y el vector nadir, en cada generación, GWASF-GA divide la población de padres e hijos en diferentes fronteras en base a los valores que dichos individuos alcanzan para una función escalarizada de logro. Son preferibles aquellas soluciones que alcanzan los menores valores de la función de logro para el utopía y el nadir, y cada uno de los vectores de pesos. Para conseguir una buena aproximación del frente óptimo de Pareto, los vectores de pesos considerados en GWASF-GA han de cumplir la siguiente propiedad: deben definir un conjunto de direcciones de proyección uniformemente distribuidas en el espacio de objetivos. De esta forma, GWASF-GA aproxima el frente óptimo de Pareto mediante la proyección simultánea del utopía y del nadir sobre dicho frente, considerando un conjunto de direcciones de proyección distribuidas de forma uniforme. Se puede decir que la convergencia del algoritmo se asegura mediante la minimización de la función de logro en cada generación, y la diversidad se mantiene a través del conjunto de direcciones de proyección que pretenden cubrir el frente óptimo de Pareto en su totalidad.

En los experimentos computacionales realizados, se ha mostrado el funcionamiento de GWASF-GA para varios problemas test con tres y cinco objetivos. Se ha comparado las poblaciones obtenidas con las proporcionadas por los algoritmos evolutivos NSGA-II y MOEA/D con respecto a la métrica hipervolumen obtenida en 30 ejecuciones independientes. En base al estudio realizado, los resultados concluyen que nuestro algoritmo ha proporcionado mejores aproximaciones del frente óptimo de Pareto que los otros dos algoritmos en la mayoría de los problemas considerados, lo cual demuestra su potencial. Pese a que no podemos asegurar que GWASF-GA sea superior que NSGA-II y MOEA/D en cualquier problema de optimización multiobjetivo, los resultados obtenidos han sido bastante prometedores.

Posibles líneas de investigación futuras pueden ser la aplicación de GWASF-GA a problemas de opti-

TABLA I

VALORES MEDIOS Y DESVIACIONES TÍPICAS OBTENIDOS PARA LA MÉTRICA HIPERVOLUMEN

Problema	Objetivos	GWASF-GA	MOEA/D	NSGA-II
DTLZ1	3	$\mathbf{8{,}01e-01_{3{,}8e-04}}$	$7{,}83e-01_{3{,}9e-04}$	$7{,}95e-01_{2{,}0e-03}$
DTLZ2	3	$\mathbf{4{,}38e-01_{1{,}8e-04}}$	$4{,}19e-01_{8{,}3e-04}$	$4{,}17e-01_{2{,}3e-03}$
DTLZ3	3	$\mathbf{4{,}38e-01_{5{,}4e-04}}$	$4{,}20e-01_{1{,}4e-03}$	$4{,}21e-01_{2{,}9e-03}$
DTLZ4	3	$\mathbf{4{,}32e-01_{3{,}6e-04}}$	$4{,}09e-01_{3{,}1e-02}$	$4{,}16e-01_{2{,}2e-03}$
DTLZ5	3	$9{,}49e-02_{5{,}7e-07}$	$9{,}38e-02_{9{,}7e-06}$	$\mathbf{9{,}53e-02_{3{,}9e-05}}$
DTLZ6	3	$9{,}58e-02_{6{,}3e-07}$	$9{,}47e-02_{1{,}4e-06}$	$\mathbf{9{,}62e-02_{4{,}2e-05}}$
DTLZ7	3	$\mathbf{3{,}15e-01_{7{,}9e-05}}$	$2{,}56e-01_{7{,}0e-04}$	$3{,}12e-01_{9{,}4e-04}$
WFG1	3	$9{,}13e-01_{3{,}1e-03}$	$7{,}35e-01_{5{,}2e-02}$	$\mathbf{9{,}13e-01_{2{,}0e-03}}$
WFG2	3	$\mathbf{9{,}19e-01_{1{,}6e-04}}$	$8{,}94e-01_{3{,}4e-03}$	$9{,}14e-01_{9{,}3e-04}$
WFG3	3	$\mathbf{3{,}29e-01_{6{,}7e-05}}$	$3{,}16e-01_{9{,}8e-05}$	$3{,}25e-01_{9{,}7e-04}$
WFG4	3	$\mathbf{4{,}36e-01_{1{,}2e-04}}$	$4{,}03e-01_{3{,}6e-03}$	$4{,}12e-01_{2{,}5e-03}$
WFG5	3	$\mathbf{3{,}91e-01_{1{,}9e-03}}$	$3{,}58e-01_{1{,}3e-03}$	$3{,}74e-01_{2{,}3e-03}$
WFG6	3	$\mathbf{4{,}33e-01_{1{,}7e-03}}$	$4{,}05e-01_{2{,}5e-03}$	$4{,}14e-01_{3{,}2e-03}$
WFG7	3	$\mathbf{4{,}36e-01_{1{,}4e-04}}$	$4{,}07e-01_{1{,}6e-03}$	$4{,}08e-01_{2{,}9e-03}$
WFG8	3	$\mathbf{3{,}88e-01_{4{,}7e-02}}$	$3{,}22e-01_{4{,}6e-02}$	$2{,}94e-01_{3{,}9e-02}$
WFG9	3	$\mathbf{4{,}31e-01_{1{,}2e-03}}$	$3{,}96e-01_{2{,}0e-03}$	$4{,}02e-01_{2{,}3e-03}$
LZ09F6	3	$3{,}53e-01_{2{,}2e-02}$	$\mathbf{3{,}98e-01_{8{,}2e-03}}$	$3{,}00e-01_{1{,}3e-02}$
CEC09UF8	3	$2{,}67e-01_{4{,}7e-02}$	$\mathbf{3{,}12e-01_{1{,}2e-02}}$	$1{,}98e-01_{5{,}2e-02}$
CEC09UF9	3	$6{,}02e-01_{5{,}9e-02}$	$\mathbf{6{,}10e-01_{5{,}8e-02}}$	$5{,}14e-01_{9{,}3e-02}$
CEC09UF10	3	$\mathbf{1{,}05e-01_{9{,}9e-02}}$	$4{,}76e-02_{2{,}2e-02}$	$2{,}80e-02_{3{,}1e-02}$
WFG1	5	$\mathbf{9{,}99e-01_{9{,}7e-05}}$	$9{,}96e-01_{1{,}9e-03}$	$9{,}99e-01_{1{,}2e-04}$
WFG2	5	$9{,}92e-01_{1{,}8e-04}$	$\mathbf{9{,}93e-01_{5{,}1e-04}}$	$9{,}92e-01_{6{,}5e-04}$
WFG3	5	$\mathbf{1{,}99e-01_{1{,}4e-05}}$	$1{,}91e-01_{6{,}3e-04}$	$1{,}89e-01_{1{,}4e-03}$
WFG4	5	$\mathbf{6{,}60e-01_{5{,}2e-04}}$	$6{,}10e-01_{2{,}8e-03}$	$5{,}71e-01_{7{,}2e-03}$
WFG5	5	$\mathbf{6{,}29e-01_{4{,}3e-04}}$	$5{,}94e-01_{2{,}2e-03}$	$5{,}57e-01_{5{,}2e-03}$
WFG6	5	$\mathbf{7{,}03e-01_{1{,}5e-03}}$	$6{,}78e-01_{1{,}1e-03}$	$6{,}16e-01_{4{,}5e-03}$
WFG7	5	$\mathbf{6{,}69e-01_{6{,}5e-04}}$	$6{,}41e-01_{8{,}9e-04}$	$5{,}79e-01_{3{,}9e-03}$
WFG8	5	$\mathbf{6{,}79e-01_{2{,}6e-03}}$	$6{,}58e-01_{2{,}2e-03}$	$5{,}37e-01_{8{,}0e-03}$
WFG9	5	$\mathbf{6{,}60e-01_{2{,}6e-03}}$	$6{,}14e-01_{2{,}2e-03}$	$5{,}75e-01_{5{,}8e-03}$

TABLA II

TEST WILCOXON DE LA MÉTRICA HIPERVOLUMEN PARA LOS PROBLEMAS CON TRES OBJETIVOS

	MOEA/D	NSGA-II
GWASF-GA	DTLZ1, DTLZ2, DTLZ3, DTLZ4, DTLZ5, DTLZ6, DTLZ7 (3 objetivos)	
	▲ ▲ ▲ ▲ ▲ ▲ ▲	▲ ▲ ▲ ▲ ▽ ▽ ▲
	WFG1, WFG2, WFG3, WFG4, WFG5, WFG6, WFG7, WFG8, WFG9 (3 objetivos)	
	▲ ▲ ▲ ▲ ▲ ▲ ▲ ▲ ▲	– ▲ ▲ ▲ ▲ ▲ ▲ ▲ ▲
	LZ09F6,CEC09UF8, CEC09UF9, CEC09UF10 (3 objetivos)	
	▽ ▽ – –	▲ ▲ ▲ –
	WFG1, WFG2, WFG3, WFG4, WFG5, WFG6, WFG7, WFG8, WFG9 (5 objetivos)	
	▲ ▽ ▲ ▲ ▲ ▲ ▲ ▲ ▲	▲ ▽ ▲ ▲ ▲ ▲ ▲ ▲ ▲

mización con más objetivos, e incluso a problemas reales. Asimismo, una posible mejora del algoritmo podría ser la reorganización de los vectores de pesos cada cierto número de generaciones, con la intención de mejorar la convergencia del algoritmo mediante la adaptación de la búsqueda de nuevas soluciones a la topología del frente óptimo de Pareto.

AGRADECIMIENTOS

Esta investigación ha sido parcialmente financiada por la Junta de Andalucía (grupos PAI SEJ-445 y SEJ-543) y por el Ministerio de Ciencia e Innovación del Gobierno de España (proyecto MTM2010-14992).

Referencias

[Alba et al., 2009] Alba, E., Blum, C., Isasi, P., Leon, C., and Gomez, J. A. (2009). *Optimization Techniques for Solving Complex Problems*. Wiley.

[Coello et al., 2007] Coello, C. A. C., Lamont, G. B., and Veldhuizen, D. A. V. (2007). *Evolutionary Algorithms for Solving Multi-Objective Problems*. Springer, New York, 2nd edition.

[Deb, 2001] Deb, K. (2001). *Multi-objective Optimization using Evolutionary Algorithms*. Wiley, Chichester.

[Deb and Miettinen, 2010] Deb, K. and Miettinen, K. (2010). Nadir point estimation using evolutionary approaches: Better accuracy and computational speed through focused search. In Ehrgott, M., Naujoks, B., Stewart, T. J., and Wallenius, J., editors, *Multiple Criteria Decision Making for Sustainable Energy and Transportation Systems*, pages 339–354. Springer.

[Deb et al., 2002a] Deb, K., Pratap, A., Agarwal, S., and Meyarivan, T. (2002a). A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197.

[Deb et al., 2002b] Deb, K., Thiele, L., Laumanns, M., and Zitzler, E. (2002b). Scalable multi-objective optimization test problems. In *Congress on Evolutionary Computation*, pages 825–830.

[Durillo and Nebro, 2011] Durillo, J. J. and Nebro, A. J. (2011). jMetal: A java framework for multi-objective optimization. *Advances in Engineering Software*, 42:760–771.

[Durillo et al., 2010] Durillo, J. J., Nebro, A. J., and Alba, E. (2010). The jMetal framework for multi-objective optimization: Design and architecture. In *IEEE Congress on Evolutionary Computation*, pages 1–8.

[Huband et al., 2007] Huband, S., Hingston, P., Barone, L., and While, L. (2007). A review of multi-objective test problems and a scalable test problem toolkit. *IEEE Transactions on Evolutionary Computation*, 10(5):477–506.

[Ishibuchi et al., 2008a] Ishibuchi, H., Tsukamoto, N., Hitotsuyanagi, Y., and Nojima, Y. (2008a). Effectiveness of scalability improvement attempts on the performance of NSGA-II for many-objective problems. In *10th Annual Conference on Genetic and Evolutionary Computation*, pages 649–656.

[Ishibuchi et al., 2008b] Ishibuchi, H., Tsukamoto, N., and Nojima, Y. (2008b). Evolutionary many-objective optimization: A short review. In *IEEE Congress on Evolutionary Computation*, pages 2419–2426.

[Li and Zhang, 2009] Li, H. and Zhang, Q. (2009). Multiobjective optimization problems with complicated Pareto sets, MOEA/D and NSGA-II. *IEEE Transactions on Evolutionary Computation*, 12(2):284–302.

[Luque et al., 2007] Luque, M., Caballero, R., Molina, J., and Ruiz, F. (2007). Equivalent information for multiobjective interactive procedures. *Management Science*, 53(1):125–134.

[Luque et al., 2015] Luque, M., Lopez-Agudo, L., and Marcenaro-Gutierrez, O. (2015). Equivalent reference points in multiobjective programming. *Expert Systems with Applications*, 42(4):2205–2212.

[MacQueen, 1967] MacQueen, J. B. (1967). some methods for classification and analysis of multivariate observations. In *5th Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, pages 281–297.

[Miettinen, 1999] Miettinen, K. (1999). *Nonlinear Multiobjective Optimization*. Kluwer Academic Publishers, Boston.

[Ruiz et al., 2014] Ruiz, A. B., Saborido, R., and Luque, M. (2014). A preference-based evolutionary algorithm for multiobjective optimization: The weighting achievement scalarizing function genetic algorithm. *Journal of Global Optimization*, in press.

[Scheffe, 1958] Scheffe, H. (1958). Experiments with mixtures. *Journal of the Royal Statistical Society, Series B*, 20:344–360.

[Scheffe, 1963] Scheffe, H. (1963). The simplex-centroid design for experiments with mixtures. *Journal of the Royal Statistical Society, Series B*, 25(2):235–263.

[Steuer, 1986] Steuer, R. E. (1986). *Multiple Criteria Optimization: Theory, Computation and Application*. John Wiley, New York.

[Wagner et al., 2013] Wagner, T., Trautmann, H., and Brockhoff, D. (2013). Preference articulation by means of the R2 indicator. In Purshouse, R. C., Fleming, P. J., Fonseca, C. M., Greco, S., and Shaw, J., editors, *Evolutionary Multi-Criterion Optimization*, volume 7811 of *Lecture Notes in Computer Science*, pages 81–95.

[Wierzbicki, 1980] Wierzbicki, A. P. (1980). The use of reference objectives in multiobjective optimization. In Fandel, G. and Gal, T., editors, *Multiple Criteria Decision Making, Theory and Applications*, volume 177 of *Lecture Notes in Economics and Mathematical Systems*, pages 468–486. Springer.

[Wierzbicki, 1986] Wierzbicki, A. P. (1986). On the completeness and constructiveness of parametric characterizations to vector optimization problems. *OR-Spectrum*, 8(2):73–87.

[Wilcoxon, 1945] Wilcoxon, F. (1945). Individual comparisons by ranking methods. *Biometrics Bulletin*, 1(6):80–83.

[Zhang and Li, 2007] Zhang, Q. and Li, H. (2007). MOEA/D: A multiobjective evolutionary algorithm based on decomposition. *IEEE Transactions on Evolutionary Computation*, 11(6):712–731.

[Zhang et al., 2008] Zhang, Q., Zhou, A., Zhao, S., Suganthan, P. N., Liu, W., and Tiwari, S. (2008). Multiobjective optimization test instances for the CEC 2009 special session and competition. *Technical Report*, (CES-487, University of Essex and Nanyang Technological University).

[Zitzler and Thiele, 1999] Zitzler, E. and Thiele, L. (1999). Multiobjective evolutionary algorithms: A comparative case study and the strength Pareto approach. *IEEE Transactions on Evolutionary Computation*, 3(4):257–271.