Revista Cubana de Ciencias Informáticas
ISSN: 2227-1899 | RNPS: 2301
Vol. 8, No. 4, Octubre-Diciembre, 2014
Págs. 86-99
http://rcci.uci.cu

Tipo de artículo: Artículo original
Temática: Inteligencia artificial
Recibido: 19/03/2014 | Aceptado: 5/06/2014

Un estudio comparativo sobre evolución diferencial auto-adaptativa en ambientes dinámicos

A comparative study on self-adaptive differential evolution in dynamic environments

Dr. C Pavel Novoa-Hernández[1], Dr. C Carlos Cruz Corona[2], Dr. C David A. Pelta[2]

[1] Dpto Lic. Matemática, Universidad de Holguín. Cuba.
[2] Departamento de Ciencias de la Computación e Inteligencia Artificial, Universidad de Granada, España. Correo-e: {carloscruz, dpelta}@decsai.ugr.es

[*] Autor para correspondencia: pnovoa@facinf.uho.edu.cu

Resumen

Un gran número de problemas de optimización reales son dinámicos, lo que significa que algunos de los elementos del modelo varían con el tiempo. Estos problemas han recibido un especial interés en los últimos años desde el punto de vista de la optimización mediante metaheurísticas. La Evolución Diferencial (DE) es una metaheurística poblacional que sobresale por su efectividad y fácil implementación. Sin embargo, como la mayoría de los paradigmas aplicados a este contexto, DE ha tenido que ser adaptada con el objetivo resolver principalmente la pérdida de diversidad en la población de soluciones. Por otro lado, la auto-adaptación es uno los enfoques menos tratados en la actualidad, y que ha mostrado ser en extremo eficiente en determinados contextos. La auto-adaptación es una técnica de control de parámetros que dota de cierta inteligencia al algoritmo, durante el proceso de búsqueda. En ese sentido, el presente trabajo investiga dos extensiones auto-adaptativas de DE en combinación con otros enfoques de diversidad existentes. Los resultados, obtenidos a partir de varios experimentos computacionales, confirman que la auto-adaptación es una técnica prometedora en este contexto.

Palabras clave: ambientes dinámicos, auto-adaptación, evolución diferencial.

Abstract:

Several real optimization problems are dynamic, meaning that some elements of their mathematical model are time varying. These problems have received a special interest in the last years from the viewpoint of metaheuristics. Differential Evolution (DE) is one of the current population-based metaheuristics with an excellent effectiveness and easy implementation. However, as similar paradigms in dynamic environments, DE has been adapted with aims of solving the diversity loss in the solution population. On the other hand, self-adaptation is one of the less used approaches in dynamic environments, despite its success in complex scenarios. Self-adaptation is a parameter control technique that gives certain intelligent behavior to the algorithm, during the search process. In that sense, the present work investigates the performance of two self-adaptive extensions of DE, which in combination with other existing diversity approaches have been applied in several dynamic scenarios. The obtained results from the computational experiments, confirm that self-adaptation is a promising technique for dynamic environments.

Keywords: *differential evolution, dynamic environments, self-adaptation.*

Revista Cubana de Ciencias Informáticas
ISSN: 2227-1899 | RNPS: 2301
Vol. 8, No. 4, Octubre-Diciembre, 2014
Págs. 86-99
http://rcci.uci.cu

Introducción

En la sociedad moderna, varios escenarios de decisión pueden modelarse como problemas dinámicos de optimización (PDOs). Estos problemas tienen como característica la variación en el tiempo de algunos de los elementos del modelo. Ejemplo de estos elementos dinámicos son la función objetivo, el sistema de restricciones, el número de dimensiones del espacio de búsqueda, entre otros. En el presente trabajo se abordarán PDOs en los que la función objetivo varía en el tiempo. En escenarios reales, esta variación de la función objetivo en el tiempo se puede interpretar como un cambio en la información relacionada con el problema en cuestión. Algunos ejemplos donde pueden aparecer este fenómeno son: la automatización del funcionamiento de los semáforos en una ciudad, el aterrizaje de una nave espacial, la distribución de bienes a clientes que aparecen o desaparecen con el tiempo, la planificación de procesos cuyas tareas que aparecen con el tiempo, entre otros.
Para los propósitos de la presente investigación, bastará definir un PDO de la siguiente forma:

$$\min f(t, \boldsymbol{x}) \tag{1}$$

donde $f: T \times \mathbb{R}^n \to \mathbb{R}$ es la función objetivo. En particular, el espacio de búsqueda viene representado por $\Omega \subset \mathbb{R}^n$, mientras que $t \in T$ es el tiempo.

Debido al importante nivel de incertidumbre de estos problemas, la aplicación de técnicas de optimización provenientes de campos como la Soft Computing parece justificarse. Precisamente, en los últimos años este tema ha recibido un especial interés expresado en el creciente número de trabajos tanto de congresos como de revistas (véase por ejemplo las revisiones (Cruz *et al.*, 2011; Nguyen *et al.*, 2012). La tendencia actual en este campo, en relación con la propuesta de nuevos métodos, es el empleo de metaheurísticas (Melián *et al.*, 2003; Talbi, 2009) poblacionales del campo de la optimización estacionaria que son adaptados producto a los retos que impone la optimización dinámica.

Específicamente, estas adaptaciones tienen como objetivo tratar dos dificultades que pueden aparecer producto a los cambios en el ambiente: la *desactualización de la información* y la *pérdida de diversidad*. La primera ocurre cuando los valores de la función objetivo (fitness) correspondientes a las soluciones del algoritmo quedan obsoletos. Por su parte, la pérdida de diversidad aparece cuando el algoritmo queda atrapado en una zona del espacio de búsqueda y es incapaz de localizar al nuevo óptimo del problema. Aunque la primera de estas dificultades puede resolverse relativamente fácil (ej. reevaluando la población de soluciones), la segunda es más difícil de tratar. En ese sentido, en los últimos años se han propuesto diferentes *enfoques para PDOs* que pueden agruparse, según (Jin y Branke, 2005), en cuatro categorías: 1) diversidad después del cambio, 2) diversidad durante la ejecución, 3) enfoques basados en memorias, y 4) enfoques multipoblacionales. Más recientemente, en (Nguyen *et al.*, 2012) se establece que otra forma de lidiar con la dinámica de los PDOs, es aplicar estrategias auto-adaptativas. La auto-adaptación es una técnica de control de parámetros que ha sido ampliamente utilizada en problemas estacionarios (Eiben *et al.*, 2007). En ambientes dinámicos existen, aunque pocos, prometedores trabajos que sugieren la aplicación de esta técnica de control de parámetros (Brest *et al.*, 2009; Novoa-Hernández *et al.*, 2013).

En este contexto, una de las metaheurísticas poblacionales que ha sido aplicado con éxito en ambientes dinámicos es la Evolución Diferencial (DE). Sin embargo, similar a lo que ocurre en otros algoritmos poblacionales, DE sufre de pérdida de diversidad y en consecuencia tiene que ser adaptado a este contexto. Aunque en la actualidad se han

Revista Cubana de Ciencias Informáticas
ISSN: 2227-1899 | RNPS: 2301
Vol. 8, No. 4, Octubre-Diciembre, 2014
Págs. 86-99
http://rcci.uci.cu

logrado importantes avances en esa dirección, se trata de un tema abierto y de constante investigación. En ese sentido, el presente trabajo tiene por objetivo: analizar empíricamente el comportamiento, en términos de precisión, de dos extensiones auto-adaptativas de DE en ambientes dinámicos. Dado que estas extensiones están diseñadas para problemas estacionarios, se hace necesario hibridarlas con algunos enfoques para PDOs. En este caso, se consideraron el enfoque multipoblacional propuesto por (Blackwell y Branke, 2006), y estrategias específicas de diversidad, como las de los trabajos (Blackwell y Branke, 2006) y (Novoa-Hernández, 2011). Nuestra hipótesis es que la auto-adaptación, en conjunto con estos enfoques, puede contribuir significativamente en la solución de PDOs complejos, como los considerados en este trabajo.

El resto de la investigación queda organizada de la forma siguiente. En lo que sigue se exponen los fundamentos teóricos de la metodología computacional desarrollada, así como la descripción de los algoritmos propuestos. Más adelante (Sec. *Resultados y discusión*), se describirán los experimentos computacionales desarrollados, los cuales concluyen con un análisis estadístico. El trabajo finaliza con las principales conclusiones y trabajos futuros derivados de la investigación.

Metodología computacional

La Evolución Diferencial es una metaheurística poblacional propuesta por (Storn y Price, 1997). De manera formal, cada individuo $\mathbf{y}_i$ de la población se define por la tupla $\mathbf{y}_i = \langle \mathbf{x}_i, f_i \rangle$. Donde $\mathbf{x}_i \in \mathbb{R}^n$ es el vector solución y $f_i = f(\mathbf{x}_i)$ el valor correspondiente de la función objetivo. Como todo algoritmo evolutivo consta de dos etapas generales: inicialización y ciclo principal. En el primero, la población es pseudo-aleatoriamente inicializada en el espacio de búsqueda (ej. siguiendo una distribución uniforme), mientras que en el ciclo principal se generan nuevos individuos (soluciones candidatas) mediante cruzamiento y mutación. La aplicación de estos operadores evolutivos es controlada por el parámetro CR. El operador de reemplazamiento es simple y no depende de una generación (iteración) del algoritmo: si el nuevo individuo es mejor (ej. según el fitness) entonces este reemplaza al individuo que le dio origen. En particular, la mutación depende básicamente del parámetro F conocido como factor de escala. Los principales pasos del esquema original de DE se muestran en la Figura 1.

1: Inicializar en Ω la población P de μ individuos;
2: **mientras** no se cumpla condición de parada **hacer**
3: **por cada** individuo $\mathbf{y}_i$ en P **hacer**
4: Crear el vector donante $\mathbf{v}_i \in \mathbb{R}^n$;
5: Seleccionar aleatoriamente $y_r, y_s, y_t \in P$;
6: Seleccionar un j aleatoriamente en $\{1, \dots, n\}$.
7: **por cada** dimensión d en $\{1, \dots, n\}$ **hacer**
8: $$v_i^d \leftarrow \begin{cases} x_r^d + F\left(x_s^d - x_t^d\right) & \textbf{si } j = d \textbf{ o } \text{rand}(0,1) \leq CR \\ x_i^d & \text{por el contrario.} \end{cases}$$
9: **fin**
10: $f_v \leftarrow f(t, \mathbf{v}_i)$
11:

Revista Cubana de Ciencias Informáticas
ISSN: 2227-1899 | RNPS: 2301
Vol. 8, No. 4, Octubre-Diciembre, 2014
Págs. 86-99
http://rcci.uci.cu

si $f_v < f_i$ **entonces**

$\mathbf{y}_i \leftarrow \langle \mathbf{v}_i, f_v \rangle$

fin

fin

fin

Figura 1. Evolución diferencial estándar.

Aquí $rand(0,1)$ es una función que devuelve números aleatorios siguiendo una distribución uniforme en el intervalo $[0{,}0;\ 1{,}0]$. Asimismo, es importante notar que los resultados de DE en un problema dado, depende principalmente del tamaño de la población, los parámetros F, CR, y la estrategia de mutación (paso 8, en la Figura 1). En varios trabajos (véase por ejemplo (Das y Suganthan, 2011; Neri y Tirronen, 2010)), se sugiere que los parámetros F y CR se establezcan en $1{,}0$ y $0{,}1$, respectivamente; el tamaño de la población (μ), un valor cercano a 10 veces el número de dimensiones del espacio de búsqueda, mientras que la estrategia de mutación depende del problema en cuestión.

Dentro de las estrategias más conocidas se encuentran las siguientes:

DE/rand/1:

$$v_i^d = x_r^d + F\left(x_s^d - x_t^d\right) \tag{2}$$

DE/cur-to-best/1:

$$y_i^d = x_i^d + F\left(x_{best}^d - x_i^d\right) + F\left(x_r^d - x_s^d\right) \tag{3}$$

DE/rand-to-best/2:

$$v_i^d = x_r^d + F\left(x_{best}^d - x_i^d\right) + F\left(x_s^d - x_t^d\right) + F\left(x_u^d - x_v^d\right) \tag{4}$$

DE/current-to-rand/1:

$$v_i^d = x_i^d + + K\left(x_r^d - x_i^d\right) + F_p\left(x_s^d - x_t^d\right) \tag{5}$$

Aquí, K y F_p son factores de escala que toman valores en tiempo de ejecución en el intervalo $[0{,}1]$. Además, x_{best}^d corresponde a la d-ésima componente de la posición del mejor individuo de la población.

En el contexto de la optimización en ambientes dinámicos, DE ha sido aplicada con éxito en varios contextos. Por ejemplo, la extensión DynDE propuesta por (Mendes y Mohai, 2005) utiliza varias poblaciones que exploran simultáneamente el espacio de búsqueda, lo que la hace muy adecuada para problemas multimodales. Curiosamente, DynDE no requiere el establecimiento de los parámetros F y CR ya que los selecciona en tiempo de ejecución en el intervalo $[0{,}1]$. Asimismo, este método incluye técnicas de generación de diversidad mediante la perturbación de uno o más soluciones de la población. DynDE fue aplicada con buenos resultados en el problema test Movimiento de

Revista Cubana de Ciencias Informáticas
ISSN: 2227-1899 | RNPS: 2301
Vol. 8, No. 4, Octubre-Diciembre, 2014
Págs. 86-99
http://rcci.uci.cu

Picos (Branke, 1999). Más recientemente, en la competición *Evolutionary Computation in Dynamic and Uncertain Environments* (CEC2009), (Brest *et al.*, 2009) proponen una extensión al algoritmo auto-adaptativo jDE que utiliza varias poblaciones y una estrategia basada en la edad de los individuos. Este algoritmo alcanzó el primer lugar en dicha competición. Por otro lado, (Angira y Santosh, 2007) reportan experiencias en entornos reales, en lo que se aplica una versión trigonométrica de DE, con el objetivo de resolver problemas dinámicos en la ingeniería química.

Extensiones auto-adaptativas de la evolución diferencial

El esquema original de DE ha sido satisfactoriamente extendido por varios autores con la intención de mejorar su robustez, esto es, su capacidad de resolver problemas con diferentes características. Dentro de estas extensiones sobresalen las que proponen la auto-adaptación de los parámetros y estrategias que influyen en el comportamiento del algoritmo. La auto-adaptación, como técnica estado-del-arte en el control de parámetros no es originaria de DE. De hecho su historia se remonta al origen de otros paradigmas evolutivos como las Estrategias Evolutivas (Schewefel, 1981), la Programación Evolutiva (Fogel, 1992), y algunas versiones de Algoritmos Genéticos de codificación real (Beyer y Deb, 2001). Básicamente, este técnica consiste en la inclusión, dentro del ciclo evolutivo, de los parámetros más importantes del algoritmo, lo que significa que estos *evolucionan* al mismo tiempo que el conjunto de soluciones (Eiben *et al.*, 2007).

En relación a DE, existen en la literatura varias propuestas interesantes que incluyen de una forma u otra auto-adaptación. Por ejemplo, SADE (Qin *et al.*, 2005) y SspDE (Pan *et al.*, 2011) aplican auto-adaptación en la selección de la estrategia de mutación, mientras que jDE (Brest *et al.*, 2006), DESAP (Teo, 2006), y SDE (Salman *et al.*, 2007) lo hacen en los parámetros F y CR. Particularmente, DESAP también auto-adapta el tamaño de la población. De manera general, la mayoría de estas extensiones superan en precisión a la versión estándar de DE en problemas estacionarios. En el presente estudio, se han seleccionado específicamente los algoritmos jDE y SspDE, debido a dos razones: por un lado estos algoritmos son, en relación con las otras variantes, los de mejores resultados; y por otro, poseen marcadas diferencias entre sí, lo cual enriquecería las comparaciones. A continuación se describen las ideas básicas de estos algoritmos.

El algoritmo jDE es muy similar al esquema original de DE, la diferencia radica en que los individuos contienen realizaciones de los parámetros F y CR (ej. $\mathbf{y}_i = \langle x_i, f_i, F_i, CR_i \rangle$). Entonces, en la generación de cada vector donante se utilizan los valores de las expresiones (V) y (VI). Si el nuevo individuo es mejor que su padre, en el reemplazamiento se tiene en cuenta también estos valores de F_i y CR_i.

$$F_v = \begin{cases} 0{,}1 + 0{,}9 \cdot rand(0,1) & si\ rand(0,1) < \tau_1 \\ F_i & por\ el\ contrario \end{cases} \quad (6)$$

$$CR_v = \begin{cases} rand(0,1) & si\ rand(0,1) < \tau_1 \\ F_i & por\ el\ contrario \end{cases} \quad (7)$$

Los parámetros τ_1 y τ_2 permanecen fijos durante la ejecución y según sus autores deben tomar el mismo valor, esto es, igual a 0.1.

Por su parte, el algoritmo SspDE es una extensión más sofisticada de DE. Los individuos vienen dados como $\mathbf{y}_i = \langle x_i, f_i, \mathbf{F}_i, \mathbf{CR}_i, \mathbf{S}_i, wp_i, \mathbf{F}_i^*, \mathbf{CR}_i^*, \mathbf{S}_i^* \rangle$. Donde $\mathbf{F}_i$ y $\mathbf{CR}_i$ son listas de posibles valores para los parámetros F_i y CR_i,

Revista Cubana de Ciencias Informáticas
ISSN: 2227-1899 | RNPS: 2301
Vol. 8, No. 4, Octubre-Diciembre, 2014
Págs. 86-99
http://rcci.uci.cu

respectivamente. Asimismo, $\mathbf{S}_i$ es un conjunto de varias estrategias de mutación, específicamente instancias de las cuatro definidas por las expresiones (2-5). Los vectores $\mathbf{F}_i^*, \mathbf{CR}_i^*$, y $\mathbf{S}_i^*$ son listas de parámetros y estrategias *ganadoras*, esto es, aquellos valores de $\mathbf{F}_i, \mathbf{CR}_i$, y $\mathbf{S}_i$ relacionados con la creación de un vector de mejor calidad que el padre, siendo estas mejoras contadas por la variable wp_i. De manera general, todos los vectores poseen una longitud predefinida conocida como *período de aprendizaje*, denotado por LP. Una vez que este período termina se crean nuevamente las listas de parámetros y estrategias a partir de las listas *ganadoras*. La idea principal de este método es el aprovechamiento de configuraciones de parámetros y estrategias que resultaron útiles en determinados momentos (ej. cada LP generaciones) para favorecer la optimización en generaciones posteriores. El lector puede encontrar más detalles de este algoritmo en el trabajo original (Pan *et al.*, 2011).

Algoritmos propuestos

Como se advirtió anteriormente, los algoritmos que serán objeto de análisis en este trabajo, son hibridaciones de: 1) las extensiones auto-adaptativas jDE y SspDE; y 2) determinados enfoques para PDOs. Específicamente, los enfoques son los siguientes:

- Enfoque multipoblacional (Blackwell y Branke, 2006)
- Enfoque quantum (diversidad durante la ejecución) (Blackwell y Branke, 2006)
- Enfoque diversidad después de los cambios (Novoa-Hernández *et al.*, 2011).

La Figura 2 muestra la plantilla general que tendrán los algoritmos, la cual es en esencia el enfoque multipoblacional de (Blackwell y Branke, 2006). Este enfoque, aunque fue propuesto en el contexto de la Optimización con Enjambres de Partículas (PSO), puede ser fácilmente reutilizado empleando otros paradigmas distintos a PSO (ej. jDE y SspDE). Básicamente, este enfoque emplea varias poblaciones independientes (sub-algoritmos), lo que lo hace muy apropiado para problemas multimodales. Las poblaciones son controladas por un operador de *exclusión* que evita que dos o más poblaciones se encuentren en la misma zona del espacio de búsqueda. Si este fuera el caso, el peor de los enjambres es reiniciado en el espacio de búsqueda. Para determinar cuando ocurre este estado, la distancia euclidiana entre las mejores soluciones de ambas poblaciones tiene que ser menor que un umbral r_{excl}.

```
1: por cada población P_i hacer
2:     Inicializar a P_i en Ω;
   fin
3: mientras no se cumpla condición de parada hacer
4:     aplicar el principio de exclusión entre poblaciones;
5:     detectar cambios en el ambiente;
6:     si no ocurre un cambio en el ambiente hacer
7:         por cada población P_i hacer
8:             Iterar a P_i según la extensión de DE considerada (ej. DE, jDE, o SspDE);
9:         fin
```

Revista Cubana de Ciencias Informáticas
ISSN: 2227-1899 | RNPS: 2301
Vol. 8, No. 4, Octubre-Diciembre, 2014
Págs. 86-99
http://rcci.uci.cu

```
10:        si no
11:            por cada población P_i hacer
                   Reevaluar todos los individuos de P_i;
               fin
           fin
       fin
```

Figura 2. Plantilla general de los algoritmos considerados.

Adicionalmente, este trabajo (Blackwell y Branke, 2006) incluye un enfoque de diversidad durante la ejecución. Dicho enfoque, se basa en la inclusión de individuos (partículas) *quantum*, las cuales se generan de manera uniforme alrededor de la mejor solución de la población. Estas partículas no poseen un comportamiento convergente, ya que se mantienen explorando en una hiperesfera de radio fijo r_{cloud}.

En relación a la estrategia de diversidad de (Novoa-Hernández *et al.*, 2011), es preciso decir que esta es, en principio, similar a la de las partículas quantum, sin embargo esta estrategia se ejecuta después de cada cambio. Formalmente, después de cada cambio la mitad de la población es reiniciada en una hiperesfera de radio r_{div} entorno a la mejor solución.

Es importante señalar algunos aspectos que sobresalen en relación con esta hibridación de DE y ambas estrategias de diversidad. Por ejemplo, cuando se utilizan partículas quantum la población de soluciones tendrá individuos de diferentes tipos (quantum y los correspondientes a la extensión de DE). Esto evidentemente afecta la velocidad de convergencia de las sub-poblaciones por las características de las estrategias de mutación (ej. por la elección aleatoria de individuos de la población). Sin embargo, puede ser un comportamiento deseable en situaciones donde se requiera mantener la diversidad (ej. problemas con frecuencia de cambio muy altas). Por tal motivo, los algoritmos propuestos basados en DE que incluyen individuos de tipo quantum tendrán una población mixta, propiciando así la interacción de sus individuos mediante la estrategia de mutación. En relación con la aplicación de la estrategia de diversidad después del cambio no existen dificultades mayores pues se trata de variar el vector solución de individuos existentes.

En la Tabla 1, se listan los algoritmos implementados. Nótese que se ha incluido además, para favorecer la comparación, el algoritmo mQSO (Blackwell y Branke, 2006) al final de la Tabla 1. Asimismo, se ha considerado al algoritmo DE estándar (presente en mR+DE y mQ+DE), como alternativa básica a las extensiones auto-adaptativas. Las extensiones que emplean el enfoque quantum están denotadas con el prefijo *mQ*, mientras que las que emplean diversidad después del cambio son identificadas por *mR*.

Tabla 1. Algoritmos propuestos.

Algoritmo	**Enfoque quantum**	**Diversidad**	**Auto-adaptación**
mR+DE		Sí	
mR+JDE		Sí	Sí
mR+SspDE		Sí	Sí
mQ+DE	Sí		
mQ+jDE	Sí		Sí

Revista Cubana de Ciencias Informáticas
ISSN: 2227-1899 | RNPS: 2301
Vol. 8, No. 4, Octubre-Diciembre, 2014
Págs. 86-99
http://rcci.uci.cu

mQ+SspDE	Sí	Sí
mQSO (Blackwell y Branke, 2006)	Sí	

Resultados y discusión

En esta sección se exponen los resultados de los experimentos realizados sobre el problema artificial Movimiento de Picos (MPB) (Branke, 1999). Dada su flexibilidad, este problema permite obtener diferentes instancias a partir de las combinaciones de sus parámetros. En la Tabla 2 se muestran las configuraciones generales utilizadas en este trabajo y que corresponden a lo que se conoce como escenario 2 de este problema.

Para medir el comportamiento de los algoritmos, se empleó el promedio del error de la mejor solución antes de cada cambio (e_{mac}). Esta medida cuantifica la precisión, en términos del error absoluto del valor de la función objetivo, justo antes de ocurrir un cambio. Formalmente se define como:

$$e_{mac} = \frac{1}{C}\sum_{c=1}^{C} |f(t_c, \mathbf{x}^*) - f(t_c, \mathbf{x}_{best})| \quad (8)$$

donde C son los cambios en el problema, mientras que t_c es la última evaluación de la función objetivo antes de la ocurrencia del cambio c. Por otro lado, $\mathbf{x}^*$ es la posición del óptimo del problema, y $\mathbf{x}_{best}$ es la mejor solución encontrada por el algoritmo.

Tabla 2. Configuración del escenario 2 del generador MPB.

Parámetro	**Configuración**
Número de óptimos (picos)	10
Dimensiones	5
Altura de los picos	$\in [30;\ 70]$
Anchura de los picos	$\in [1;\ 12]$
Frecuencia de los cambios (Δe)	$\in \{1000;\ 5000;\ 10000\}$
Severidad de los cambios (sev)	$\in \{1;\ 5;10\}$
Factor de correlación de los cambios	0,0

Los experimentos fueron divididos en dos grupos con la intención de analizar los algoritmos en problemas con características diferentes. En ese sentido, el factor que se tomó en cuenta fue el tipo de función que define a los picos: en el primer grupo se utilizarán funciones unimodales, mientras que en el segundo multimodales. La Tabla 3 muestra las definiciones de las funciones seleccionadas. En ambos grupos de experimentos por cada función pico se consideraron varios valores para la frecuencia de los cambios: $\Delta e = \{1000, 5000, 10000\}$ y la severidad $sev = \{1;\ 5;\ 10\}$. Nótese que a partir de la combinación de estos valores y las funciones pico empleadas, se obtienen 54 instancias de problemas diferentes.

Tabla 3. Funciones pico empleadas en el escenario 2 del generador MPB (Branke, 1999).

Función	**Fórmula**	**Ω**

Revista Cubana de Ciencias Informáticas
ISSN: 2227-1899 | RNPS: 2301
Vol. 8, No. 4, Octubre-Diciembre, 2014
Págs. 86-99
http://rcci.uci.cu

Unimodal	*Cone*	$f(\mathbf{x}) = \sqrt{\sum_{i=1}^{n} x_i^2}$	$[0;\ 100]^5$
	Sphere	$f(\mathbf{x}) = \sum_{i=1}^{n} x_i^2$	$[0;\ 100]^5$
	Schwefel	$f(\mathbf{x}) = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$	$[0;\ 100]^5$
Multimodal	*Rastrigin*	$f(\mathbf{x}) = \sum_{i=1}^{n} (x_i^2 - 10\cos 2\pi x_i + 10)$	$[-5{,}12;\ 5{,}12]^5$
	Ackley	$f(\mathbf{x}) = -20 \exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n} \cos 2\pi x_i\right) + 20 + \exp(1)$	$[-32;\ 32]^5$
	Griewank	$f(\mathbf{x}) = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	$[-32;\ 32]^5$

Todos los algoritmos emplearán 10 sub-poblaciones de 10 individuos cada una, en particular, los algoritmos que utilizan individuos quantum tendrán 5 de este tipo. Además, se empleó la siguiente configuración general: $r_{excl} = r_{div} = 0{,}3 \cdot \Omega_{ext}$, mientras que $r_{cloud} = 0{,}01 \cdot \Omega_{ext}$. Donde Ω_{ext} es la extensión del espacio de búsqueda. Las extensiones basadas en DE y jDE utilizaron DE/Rand/1 (Expresión (I)) como estrategia de mutación. De manera general se consideraron 20 ejecuciones por cada par problema algoritmo con semillas aleatorias diferentes. Para todos los problemas el número de cambios (C) se estableció en 50, de manera que cada ejecución termina cuando el algoritmo consume $50 \cdot \Delta e$ evaluaciones de la función objetivo.

Los resultados de los experimentos, en términos del promedio del error antes del cambio, y del error estándar, se muestran en las tablas de los Anexos A y B. En el primero, se resume los resultados de los algoritmos en instancias de problema con funciones pico unimodales, mientras que en el segundo se muestran los relacionados con funciones pico multimodales. No obstante la utilidad de esta información descriptiva, se realizaron pruebas estadísticas tomando como base estos resultados, con el objetivo de comparar formalmente a los algoritmos. En particular, hemos seguido la metodología propuesta en (García *et al.*, 2009), la cual sugiere el empleo de pruebas no paramétricas.

El análisis lo hemos organizado en tres grupos, basados en el tipo de función empleada por las instancias de problema. En los tres casos las pruebas de Friedman e Iman-Davenport dieron p-valores por debajo 0,05. Lo que indica que existen diferencias significativas a nivel de grupo. Como consecuencia, se procedió con pruebas post-hoc, en específico se aplicó la prueba de Holm. La idea de este análisis post-hoc es determinar si el mejor de los algoritmos en cada caso, es diferente o no en relación al resto.

La Tabla 4 muestra los rangos (posiciones) medias de los algoritmos en los diferentes grupos, según la prueba de Friedman. Es importante aclarar que un valor bajo del rango medio indica que el algoritmo correspondiente posee un buen comportamiento, en términos del error antes del cambio. Por ejemplo, nótese que en el caso de las funciones unimodales, el mejor algoritmos es el de la literatura, esto es mQSO, el cual obtiene un rango de 1,0. Esta superioridad es significativa en relación al resto de los algoritmos, según lo demuestra la prueba de Holm. Esto último está denotado por el símbolo (*) en la Tabla 4. Asimismo, el peor de los algoritmos en este grupo es mR+DE, con un rango medio de 7,02. Precisamente en esta clase de problema, los algoritmos auto-adaptativos, no muestran buenos resultados en sentido general. A excepción de los que presentan el enfoque quantum, el resto (mR+jDE y mR+SspDE) poseen resultados inferiores incluso al mQ+DE, el cual no posee auto-adaptación. Resulta particularmente clara la ventaja que reporta incorporar el enfoque quantum cuando se resuelve este tipo de problema.

Revista Cubana de Ciencias Informáticas
ISSN: 2227-1899 | RNPS: 2301
Vol. 8, No. 4, Octubre-Diciembre, 2014
Págs. 86-99
http://rcci.uci.cu

Por otro lado, como se aprecia en la columna relacionada con las funciones multimodales, el algoritmo de mejores resultados es mQ+SspDE. Como sucede en el caso anterior, esta superioridad es significativa en relación al resto de los métodos. Nótese que el algoritmo mQSO, esta vez obtiene un rango medio de 5,12, siendo uno de los de peor resultado. Este tipo de problema resulta más complejo por la naturaleza de las funciones, se requiere por tanto de una mayor generación de diversidad por parte de los algoritmos. Por tal motivo, los algoritmos que incluyen la estrategia de diversidad después de los cambios, mejoran sus resultados en relación al obtenido en el grupo de funciones unimodales. La gran excepción de los algoritmos auto-adaptativos es mQ+jDE, nótese que es el de peores resultados, a juzgar por su rango medio igual a 6,09.

Tabla 4. Rangos medios de los algoritmos según la prueba de Friedman.

Algoritmo	**Funciones unimodales**	**Funciones multimodales**	**Todas las funciones**
mR+DE	7,02	4,98	5,99
mR+jDE	5,04	3,05	4,04
mR+SspDE	5,96	3,68	4,82
mQ+DE	3,96	3,11	3,53
mQ+jDE	2,16	6,09	4,12
mQ+SspDE	2,87	**1,94(*)**	**2,40(*)**
mQSO (Blackwell y Branke, 2006)	**1,00(*)**	5,12	**3,06(*)**

Los valores marcados con (*) son significativamente diferentes al resto, según la prueba post-hoc de Holm.

Finalmente, si se tiene en cuenta todas las funciones (última columna de la Tabla 4), se puede ver que el mejor algoritmo es mQ+SspDE. Sin embargo, esta superioridad es compartida con el algoritmo de la literatura mQSO, según la prueba de Holm. Por su parte, los de peores resultados son los que presentan la estrategia de diversidad, donde se incluyen algoritmos auto-adaptativos. De manera que se puede concluir que la auto-adaptación, en específico la presentada por el algoritmo SspDE, puede ser beneficiosa en una amplia gama de escenarios dinámicos. Es importante notar que este éxito no se debe exclusivamente al paradigma SspDE, sino también al enfoque quantum con el que se le ha combinado. Este último análisis puede verse mejor en el gráfico de barras de la Figura 3.

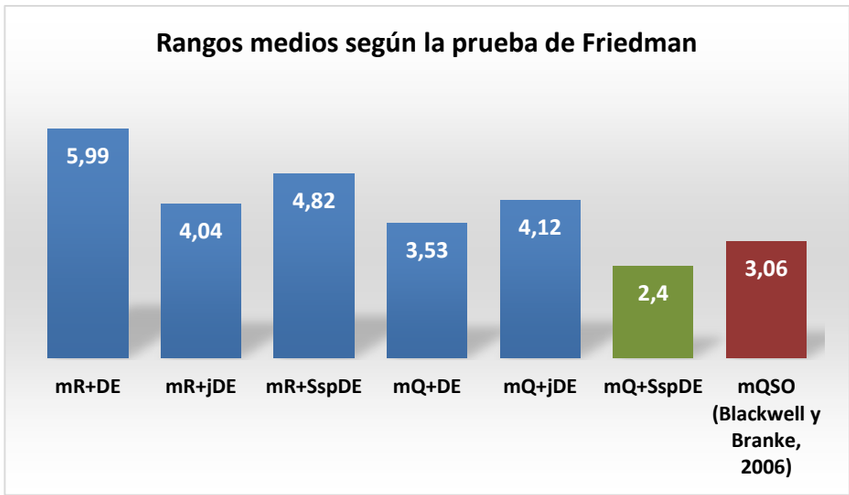

Figura 3. Rangos medios de los algoritmos en todas las instancias de s problema.

Conclusiones

En este trabajo se analizó el impacto de la auto-adaptación en la evolución diferencial, en el contexto de la resolución de problemas dinámicos de optimización. Específicamente, se consideraron dos variantes auto-adaptativas de dicha metaheurística, que fueron hibridadas con el enfoque multipoblacional de (Blackwell y Branke, 2006), y otros dos, relacionados con la generación de diversidad: enfoque quantum (Blackwell y Branke, 2006), y diversidad después de los cambios (Novoa-Hernández *et al.*, 2011). A partir de los resultados en diferentes instancias de problemas, se puede concluir que los beneficios de la auto-adaptación son en cierta medida limitados. Aunque el algoritmo más robusto resultó ser precisamente uno con auto-adaptación, esta superioridad no fue significativa en relación al exponente de la literatura. Asimismo, dicha superioridad se debe también en parte al enfoque de diversidad empleado. No obstante, estos resultados indican que la auto-adaptación, en combinación con otros enfoques de adaptación, deriva en algoritmos más precisos, al menos para los problemas y medida considerados.

Una alternativa a esta forma de aplicar auto-adaptación en ambientes dinámicos es, en lugar de emplear un paradigma auto-adaptativo (como jDE o SspDE), es incluir algún modelo auto-adaptativo directamente a enfoques específicos para PDO. Nuestros trabajos futuros estarán enfocados a esta cuestión.

Agradecimientos

Los autores agradecen a los revisores del artículo por sus sugerencias y correcciones, los cuales contribuyeron en la calidad del mismo.

C. Cruz y D. Pelta agradecen el apoyo del Ministerio de Economía y Competitividad, España, a través del proyecto TIN2011-27696-C02-01; y a la Consejería de Innovación, Ciencia y Empresa, Junta de Andalucía, a través del proyecto P11-TIC-8001 (incluyendo fondos FEDER), y a la Universidad de Granada a través del proyecto GENIL-PYR-2014-9.

Referencias

– ANGIRA R.; SANTOSH, A. Optimization of dynamic systems: a trigonometric differential evolution approach. Comput. Chem. Eng., 2007, 31:1055-1063.
– BEYER, H.-G.; DEB, K. On self-adaptive features in real-parameter evolutionary algorithms. IEEE Transactions on Evolutionary Computation, 2001, 5(3): p. 250-270.
– BLACKWELL, T., BRANKE, J. Multiswarms, exclusion, and anti-convergence in dynamic environments. IEEE Transactions on Evolutionary Computation, 2006, 10(4): p. 459-472.
– BRANKE, J. Memory enhanced evolutionary algorithms for changing optimization problems. Proceedings of the Congress on Evolutionary Computation, volume 3, pages 1875-1882, Mayower Hotel, Washington D.C., USA, IEEE Press , 1999, 6-9.
– BREST, J.; GREINER, S.; BOSKOVIC, M.; MERNIK, B.; ZUMER, V. Self-adapting control parameters in differential evolution: a comparative study on numerical benchmark problems. IEEE Transactions on Evolutionary Computation, 2006, 10(6): p. 646-657.
– BREST J, ZAMUDA A, BOSKOVIC B, MAUCEC M.S, ZUMER V Dynamic optimization using self-adaptive differential evolution. Proceedings of the Eleventh conference on Congress on Evolutionary Computation, CEC'2009, IEEE Press, Piscataway, NJ, USA, 2009, p. 415-422.
– CRUZ, C.; GONZÁLEZ, J. R.; PELTA, D. Optimization in dynamic environments: a survey on problems, methods and measures. Soft Computing, 2011, 15(7): p. 1427-1448.

- DAS, S.; SUGANTHAN, P. N. Differential evolution a survey of the state-of-the-art. IEEE Transactions on Evolutionary Computation, 2011, 15(1): p. 4-31.
- EIBEN, A.; MICHALEWICZ, Z.; SCHOENAUER, M.; SMITH, J. Parameter control in evolutionary algorithms. Parameter Setting in Evolutionary Algorithms, volume 54 of Studies in Computational Intelligence, 2007, p. 19-46. Springer Berlin / Heidelberg.
- FOGEL, D.B. An analysis of evolutionary programming. First Annual Conference on Evolutionary Programming, 1992, p 43-51.
- GARCIA, S.; MOLINA, D.; LOZANO, M.; HERRERA, F. A study on the use of non-parametric tests for analyzing the evolutionary algorithms behaviour: a case study on the cec2005 special session on real parameter optimization. J Heuristics, 2009, 15: p. 617-644.
- LI, C.; YANG, S.; NGUYEN, T. T.; YU, E. L.; YAO, X.; JIN, Y.; BEYER, H.-G. & SUGANTHAN, P. N. Benchmark Generator for CEC'2009 Competition on Dynamic Optimization. Department of Computer Science, University of Leicester, U.K., 2008.
- MELIÁN, B.; MORENO PÉREZ, J.; MORENO VEGA, J. Metaheurísticas: Una visión global. Revista Iberoamericana de Inteligencia Artificial, 2003, 19: p. 7-28.
- MENDES, R.; MOHAIS, A. S. DynDE: A differential evolution for dynamic optimization problems. Proceedings of the IEEE Congress on Evolutionary Computation, 2005, 2, p. 2808-2815.
- NERI, F.; TIRRONEN, V. Recent advances in differential evolution: a survey and experimental analysis. Artificial Intelligence Review, 2010, 33: p. 61-106.
- NGUYEN, T. T.; YANG, S.; BRANKE, J. Evolutionary dynamic optimization: A survey of the state of the art. Swarm and Evolutionary Computation 2012, 6: p. 1-24.
- NOVOA-HERNÁNDEZ, P.; CRUZ CORONA, C.; PELTA, D. A. Self-adaptive, multipopulation differential evolution in dynamic environments. Soft Computing, 2013, 17(10): p. 1861-1881.
- NOVOA-HERNÁNDEZ, P.; CRUZ CORONA, C.; PELTA, D. Efficient multi-swarm PSO algorithms for dynamic environments. Memetic Computing, Springer Berlin / Heidelberg, 2011, 3(3): p. 163-174.
- PAN, Q.-K.; SUGANTHAN, P.; WANG, L.; GAO, L.; MALLIPEDDI, R. A differential evolution algorithm with self-adapting strategy and control parameters. Computers & Operations Research, 2011, 38: p. 394-408.
- QIN, A.; SUGANTHAN, P. Self-adaptive differential evolution algorithm for numerical optimization. IEEE Congress on evolutionary computation, CEC2005, 2005, p. 1785-1791.
- SALMAN, A.; ENGELBRECHT, A.; OMRAN, M. Empirical analysis of self-adaptive differential evolution. European Journal of Operational Research, 2007, 183(2):785-804.
- SCHWEFEL, H.-P. Numerical Optimization of Computer Models John Wiley, Chichester, UK, 1981.
- SMITH, J. Self-adaptation in evolutionary algorithms for combinatorial optimisation. Studies in Computational Intelligence, 2008, 136: p. 31-57.
- STORN, R.; PRICE, K. Differential evolution-a simple and efficient heuristic for global optimization over continuous spaces. Journal of Global Optimization, 1997, 11:34-359.
- TALBI, E.-G. Metaheuristics: from design to implementation. John Wiley and Sons, 2009, 593 p.
- TEO, J. Exploring dynamic self-adaptive populations in differential evolution. Soft Computing, 2006, 10(8):673-686.

Revista Cubana de Ciencias Informáticas
ISSN: 2227-1899 | RNPS: 2301
Vol. 8, No. 4, Octubre-Diciembre, 2014
Págs. 86-99
http://rcci.uci.cu

Anexos

A. Resultados de los algoritmos en instancias de problema con funciones pico multimodales. La tabla muestra el promedio del error de la mejor solución antes del cambio ± error estándar.

Función	Δe	$sev.$	mR+DE	mR+jDE	mR+SspDE	mQ+DE	mQ+jDE	mQ+SspDE	mQSO
Cone	1000	1	17,68±0,40	10,07±0,30	12,64±0,33	5,08±0,22	4,58±0,17	4,84±0,16	**3,68±0,07**
		5	22,94±0,36	15,19±0,21	17,85±0,17	6,90±0,26	5,76±0,10	6,69±0,28	**4,63±0,14**
		10	26,25±0,31	19,37±0,22	22,21±0,30	13,24±0,31	11,65±0,24	13,16±0,31	**5,89±0,11**
	5000	1	6,94±0,29	3,57±0,10	4,81±0,25	1,79±0,08	1,51±0,09	1,37±0,07	**0,97±0,09**
		5	8,68±0,30	4,98±0,21	6,21±0,27	2,38±0,09	1,68±0,11	1,82±0,08	**1,36±0,12**
		10	9,87±0,21	5,62±0,21	8,04±0,18	3,04±0,14	1,90±0,09	2,42±0,14	**1,54±0,09**
	10000	1	6,37±0,27	3,22±0,07	3,24±0,12	1,26±0,05	0,80±0,07	0,87±0,11	**0,43±0,06**
		5	7,12±0,26	4,09±0,09	3,70±0,16	1,56±0,06	0,90±0,09	1,13±0,12	**0,66±0,08**
		10	7,87±0,30	4,06±0,18	5,59±0,22	2,20±0,09	1,04±0,07	1,40±0,11	**0,88±0,08**
Sphere	1000	1	40,92±1,46	16,88±0,79	20,54±1,03	7,26±0,84	5,21±0,68	5,04±0,63	**2,32±0,33**
		5	72,94±2,14	32,37±1,30	37,47±1,33	9,87±0,74	6,71±0,64	7,78±0,61	**3,20±0,27**
		10	105,89±2,33	51,33±1,28	61,73±1,23	23,35±1,00	19,34±0,84	22,38±0,73	**5,46±0,30**
	5000	1	5,93±0,40	2,34±0,17	3,10±0,25	0,44±0,04	0,18±0,04	0,19±0,03	**0,06±0,02**
		5	7,20±0,47	3,75±0,18	5,42±0,23	0,93±0,06	0,22±0,03	0,33±0,05	**0,08±0,02**
		10	10,48±0,64	3,94±0,20	7,83±0,30	1,04±0,05	0,39±0,04	0,69±0,09	**0,18±0,04**
	10000	1	4,74±0,28	1,77±0,11	2,29±0,22	0,27±0,04	0,06±0,03	0,07±0,02	**0,00±0,00**
		5	5,58±0,30	2,52±0,22	3,79±0,27	0,54±0,05	0,06±0,03	0,15±0,04	**0,01±0,01**
		10	8,04±0,35	2,99±0,18	4,97±0,28	0,59±0,05	0,05±0,02	0,20±0,04	**0,02±0,01**
Schwefel	1000	1	24,60±0,46	12,90±0,23	16,61±0,28	7,99±0,32	6,67±0,24	7,22±0,29	**5,21±0,10**
		5	33,73±0,48	21,40±0,24	24,91±0,24	12,33±0,26	10,83±0,23	11,78±0,34	**7,46±0,13**
		10	39,83±0,50	28,84±0,35	31,73±0,30	23,32±0,28	23,12±0,38	23,66±0,34	**12,20±0,23**
	5000	1	8,33±0,30	4,34±0,18	6,17±0,21	2,90±0,11	2,00±0,13	1,95±0,13	**1,55±0,11**
		5	10,49±0,32	5,75±0,24	7,44±0,25	3,81±0,11	2,56±0,12	2,56±0,07	**1,76±0,11**
		10	12,33±0,29	6,18±0,17	9,23±0,26	4,50±0,18	2,89±0,10	4,00±0,15	**2,04±0,12**
	10000	1	7,85±0,24	4,06±0,08	4,78±0,21	1,98±0,10	1,38±0,13	1,33±0,12	**0,66±0,06**
		5	8,85±0,14	4,55±0,15	5,32±0,25	2,63±0,11	1,40±0,11	1,49±0,12	**0,98±0,11**
		10	10,18±0,26	4,32±0,17	6,75±0,21	3,02±0,11	1,57±0,10	1,73±0,11	**0,81±0,07**

Los valores en negrita corresponden al mejor algoritmo.

Revista Cubana de Ciencias Informáticas
ISSN: 2227-1899 | RNPS: 2301
Vol. 8, No. 4, Octubre-Diciembre, 2014
Págs. 86-99
http://rcci.uci.cu

B. Resultados de los algoritmos en instancias de problema con funciones pico multimodales. La tabla muestra el promedio del error de la mejor solución antes del cambio ± error estándar.

Función	Δe	$sev.$	mR+DE	mR+jDE	mR+SspDE	mQ+DE	mQ+jDE	mQ+SspDE	mQSO
Griewank	1000	1	0,40±0,00	0,36±0,00	0,36±0,00	0,18±0,01	0,27±0,01	**0,14±0,00**	0,24±0,01
		5	0,45±0,00	0,40±0,00	0,39±0,00	0,17±0,00	0,24±0,01	**0,15±0,00**	0,19±0,01
		10	0,46±0,00	0,40±0,00	0,40±0,00	0,18±0,00	0,26±0,01	**0,15±0,00**	0,21±0,01
	5000	1	0,13±0,00	0,12±0,00	0,14±0,00	0,13±0,01	0,19±0,01	**0,05±0,00**	0,20±0,01
		5	0,16±0,00	0,15±0,00	0,16±0,00	0,09±0,00	0,19±0,01	**0,06±0,00**	0,18±0,01
		10	0,16±0,00	0,14±0,00	0,16±0,00	0,09±0,00	0,21±0,01	**0,06±0,00**	0,19±0,01
	10000	1	0,06±0,00	0,06±0,00	0,07±0,00	0,12±0,01	0,19±0,01	**0,04±0,00**	0,19±0,01
		5	0,07±0,00	0,07±0,00	0,09±0,00	0,08±0,00	0,19±0,01	**0,04±0,00**	0,18±0,01
		10	0,07±0,00	0,07±0,00	0,09±0,00	0,09±0,00	0,21±0,01	**0,04±0,00**	0,19±0,01
Ackley	1000	1	7,78±0,10	4,68±0,11	5,52±0,10	4,71±0,26	5,56±0,32	**3,62±0,13**	3,85±0,22
		5	8,52±0,08	**6,05±0,07**	6,56±0,07	6,85±0,20	10,77±0,35	6,14±0,08	7,28±0,43
		10	9,22±0,07	**7,25±0,07**	7,63±0,06	9,87±0,15	15,41±0,18	7,93±0,08	10,74±0,49
	5000	1	1,95±0,06	0,79±0,03	0,69±0,02	0,66±0,06	1,27±0,17	**0,40±0,03**	1,01±0,22
		5	2,60±0,05	1,48±0,05	1,24±0,05	**1,13±0,04**	5,46±0,38	1,96±0,07	4,99±0,30
		10	2,99±0,05	1,59±0,06	**1,42±0,06**	2,17±0,10	13,11±0,30	2,70±0,08	12,01±0,41
	10000	1	1,52±0,05	0,86±0,04	0,43±0,03	0,24±0,04	0,63±0,13	**0,11±0,01**	0,49±0,13
		5	2,11±0,06	1,40±0,05	0,72±0,05	**0,45±0,05**	3,49±0,28	1,58±0,05	4,97±0,26
		10	1,94±0,06	1,39±0,06	0,90±0,03	**0,62±0,05**	12,34±0,39	1,95±0,06	11,49±0,30
Rastrigin	1000	1	10,53±0,20	9,27±0,20	10,90±0,16	5,87±0,31	8,74±0,47	**4,72±0,20**	6,78±0,37
		5	16,78±0,15	14,73±0,12	15,54±0,14	7,12±0,24	8,94±0,54	**5,98±0,12**	7,54±0,28
		10	17,45±0,13	15,70±0,15	16,66±0,14	9,52±0,18	14,96±0,42	**7,65±0,07**	10,39±0,25
	5000	1	2,37±0,12	**1,29±0,05**	1,92±0,08	4,24±0,25	8,63±0,47	2,37±0,11	6,35±0,50
		5	5,06±0,12	3,35±0,08	4,77±0,09	4,13±0,26	7,01±0,38	**2,54±0,07**	5,43±0,26
		10	5,70±0,08	3,88±0,08	5,49±0,08	6,36±0,20	12,67±0,47	**3,36±0,07**	10,74±0,39
	10000	1	1,60±0,06	**0,93±0,04**	0,78±0,04	4,25±0,26	8,17±0,54	1,69±0,08	5,87±0,40
		5	2,71±0,07	**1,17±0,05**	1,79±0,08	3,53±0,19	7,40±0,37	1,92±0,07	5,00±0,33
		10	2,88±0,04	**1,42±0,06**	2,07±0,06	6,05±0,26	12,88±0,53	2,40±0,04	10,15±0,34

Los valores en negrita corresponden al mejor algoritmo.