UNIVERSIDAD DISTRITAL
FRANCISCO JOSÉ DE CALDAS

Revista Científica

http://revistas.udistrital.edu.co/ojs/index.php/revcie/index

CIDC

Artículo de investigación científica y tecnológica

Algoritmos evolutivos guiados por redes complejas libres de escala

Evolutionary Algorithms Guided by Scale-Free Complex Networks

Os algoritmos evolutivos guiados por redes complexas sem escala

José-Miguel Llanos-Mosquera [1]
Gerardo-Luis Muriel-López [2]
Joshua-David Triana-Madrid [3]
Víctor-Andrés Bucheli-Guerrero [4]

Recibido: mayo de 2021 **Aceptado:** enero de 2022

Para citar este artículo: Llanos-Mosquera, J. M., Muriel-López, G. L., Triana-Madrid, J. V. y Bucheli-Guerrero V. A. (2022). Algoritmos evolutivos guiados por redes complejas libres de escala. *Revista Científica*, *44*(2), 228-241. https://doi.org/10.14483/23448350.18039

Resumen

Los algoritmos de computación evolutiva permiten solucionar problemas de optimización a partir de iteraciones y etapas definidas. Una de las técnicas más utilizadas para este tipo de problemas es la evolución diferencial, que contiene propiedades de redes complejas de pequeño mundo, cuyo estudio es importante por los resultados que generan a los problemas de optimización. Teniendo en cuenta los resultados obtenidos en trabajos previos, en los que se propone un algoritmo evolutivo guiado por redes complejas de pequeño mundo, se define una propuesta que incluye redes complejas libres de escala, con el fin de validar los promedios generados por las redes complejas frente a los resultados presentados por el algoritmo evolutivo tradicional. Se definió un experimento que permite evaluar el desempeño del modelo propuesto y el del algoritmo evolutivo a través de indicadores estadísticos. También se utilizaron cuatro problemas de optimización (Ackley, Beale, Camel y Sphere) para evaluar la hipótesis en el modelo propuesto, su convergencia y la disminución de tiempos de ejecución frente al modelo base. Se observó que las redes complejas libres de escala generan mejores promedios que el algoritmo evolutivo tradicional y las redes complejas de pequeño mundo porque utilizan un mecanismo de conexión preferencial entre sus nodos y guían la combinación de individuos (soluciones), mejorando la tasa de convergencia y el rendimiento del algoritmo evolutivo en general.

Palabras clave: computación evolutiva tradicional; dinámica de la población; modelo Barabási; redes complejas; red sin escala.

1. Universidad del Valle, Cali, Valle del Cauca, Colombia. Contacto: jose.llanos@correounivalle.edu.co
2. Universidad del Valle, Cali, Valle del Cauca, Colombia. Contacto: gerardo.muriel@correounivalle.edu.co
3. Universidad del Valle, Cali, Valle del Cauca, Colombia. Contacto: joshua.triana@correounivalle.edu.co
4. Universidad del Valle, Cali, Valle del Cauca, Colombia. Contacto: victor.bucheli@correounivalle.edu.co

Abstract
Evolutionary computation algorithms allow solving optimization problems through defined iterations and stages. One of the most commonly employed techniques for this type of problem is differential evolution, which contains properties of small-world complex networks, whose study is important because of the results they generate for optimization problems. Considering the results obtained in previous works, which propose an evolutionary algorithm guided by complex small-world networks, a proposal is defined which contains complex scale-free networks, with the purpose of validating the averages generated by complex networks against the results obtained by the traditional evolutionary algorithm. An experiment was defined which allows evaluating the performance of the proposed model and that of the evolutionary algorithm by means of statistic indicators. Four optimization problems (Ackley, Beale, Camel, and Sphere) were also used to evaluate the hypothesis in the proposed model, its convergence, and the reduction of execution times compared to the base model. It was observed that the scale-free complex networks generated better averages than the traditional evolutionary algorithm and the small-world networks because they use a connection preferential mechanism between their nodes and guide the combination of individuals (solutions), thus improving the convergence rate and the performance of the evolutionary algorithm in general.
Keywords: Barabasi model; complex networks; population dynamics; scale-free network; traditional evolutionary computation.

Resumo
Algoritmos de computação evolucionária permitem resolver problemas de otimização com base em iterações e estágios definidos. Uma das técnicas mais utilizadas para este tipo de problema é a evolução diferencial, ela contém propriedades de redes complexas de pequeno mundo que são importantes para estudar, devido aos resultados gerados por problemas de otimização. Levando em consideração os resultados obtidos em trabalhos anteriores onde é proposto um algoritmo evolutivo guiado por redes complexas de pequenos mundos, é definida uma proposta que inclui redes complexas livres de escala, a fim de validar as médias geradas por redes complexas em relação aos resultados apresentados pelas redes tradicionais. algoritmo evolutivo. Foi definido um experimento que permite avaliar o desempenho do modelo proposto e do algoritmo evolutivo por meio de indicadores estatísticos. Quatro problemas de otimização (Ackley, Beale, Camel e Sphere) também foram utilizados para avaliar a hipótese no modelo proposto, sua convergência e diminuição dos tempos de execução em relação ao modelo base. Observou-se que redes complexas livres de escala geram melhores médias que o algoritmo evolucionário tradicional e redes complexas de mundos pequenos, pois utiliza um mecanismo de conexão preferencial entre seus nós, orienta a combinação de indivíduos (soluções), melhorando a taxa de convergência e a desempenho do algoritmo evolutivo em geral.
Palavras-chaves: computação evolucionária tradicional; dinâmica da população; modelo Barabási; rede sem escala; redes complexas.

Introducción

El algoritmo evolutivo es una estrategia heurística que permite resolver problemas de optimización a partir de la iteración de tres etapas (Michalewicz, 1996). En la evaluación, la población (soluciones) se genera aleatoriamente y luego se compara con sus valores fitness. En la etapa de selección, se mantienen las soluciones con mayor fitness, y en la etapa de reproducción se generan nuevos hijos con poca probabilidad de cambios (etapa de mutación).

Los algoritmos de computación evolutiva se introdujeron por primera vez en 1956 (Friedman) y 1958 (Bremermann), convirtiéndose en una de las técnicas más utilizadas para los problemas de optimización. Algunos de los algoritmos evolutivos más utilizados son: algoritmos genéticos (GA), optimización de enjambre de partículas (PSO), colonia de hormigas (AC), recocido simulado (SA), evolución diferencial (DE), algoritmos inmunes (IA), colonia de abejas artificial (ABC), algoritmo de luciérnaga (FA) y evolución diferencial (DE). Las aplicaciones

DE incluyen optimización de funciones para un solo objetivo y múltiples objetivos, entrenamiento de redes neuronales, agrupamiento y clasificación de microarrays para ADN (Plagianakos, Tasoulis y Vrahatis, 2008). En la investigación de Wang y Sobey (2020), los autores revisan las aplicaciones de los algoritmos evolutivos, específicamente los genéticos, encontrando que el 56% de los estudios aplicados utilizan este método.

En cuanto a los algoritmos de optimización, es importante evitar un mínimo local y una convergencia rápida, para mejorar el rendimiento del método. En Triana, Bucheli y García (2020b) se comprobó que, en una estrategia basada en la selección de individuos guiados por redes complejas de pequeño mundo, se puede superar la estrategia original del algoritmo de DE propuesto por Storn (1996). Otros estudios (Zelinka *et al.*, 2011, 2010, 2014; Zelinka, 2011) encontraron relación entre las redes complejas y los algoritmos evolutivos. En Zelinka, Tomaszek y Snasel (2015) se analizaron las propiedades de la red desde el punto de vista matemático, descubriéndose redes complejas para DE.

Dos redes complejas conocidas son redes de pequeño mundo y redes libres de escala (Sheng *et al.*, 2016); es fundamental estudiar ambos tipos de redes cuando se aplican en algoritmos evolutivos. Metlicka y Davendra (2017) estudiaron el uso de redes complejas con el algoritmo Firefly para evitar la convergencia en mínimos locales. Mientras que Kadavy *et al.* (2017) estudiaron la dinámica del algoritmo Fireworks con redes complejas en el problema de optimización de parámetros y Yinghui *et al.* (2018) propusieron un modelo dinámico de redes complejas en el que se evidencia que las propiedades de este tipo de redes cambian de iteración a iteración, por lo tanto, es difícil estudiar las relaciones entre nodos. Para resolver otros problemas de optimización como el vendedor viajero, en Triana, Bucheli y García (2020a) se propone un algoritmo evolutivo guiado por redes complejas de pequeño mundo. Asimismo, en Pizzuti (2018) se presentó una búsqueda bibliográfica sobre el uso de algoritmos evolutivos en la detección de grupos en series de datos que se comportan como redes complejas, esto permitió identificar la relación entre las dos áreas de investigación.

Con base en el trabajo de Triana, Bucheli y García (2020a), se presenta un modelo basado en redes complejas libre de escala, que pretende mejorar la tasa de convergencia y el rendimiento del algoritmo de computación evolutiva tradicional y las redes complejas de pequeño mundo, a partir de diferentes experimentos que definen el número de repeticiones, la razón de mutación y el número de generaciones.

El documento está organizado de la siguiente forma, en la sección de metodología se explican la propuesta y el enfoque que se definieron para evaluar el desempeño. Luego en los resultados se presentan los experimentos que se llevaron a cabo para evaluar el algoritmo evolutivo en los problemas de optimización. Al final se encuentran las conclusiones.

Metodología

Existen dos enfoques para evaluar qué tan óptimo es un algoritmo evolutivo: uno teórico y el otro empírico (Fogel, 2005). El enfoque teórico estudia el rigor matemático del algoritmo y el empírico estudia el desempeño con base en indicadores estadísticos. La metodología propuesta en este trabajo sigue un enfoque empírico basado en Triana, Bucheli y García (2020b), en el que la relación entre las soluciones y la mutación del algoritmo propuesto está guiada por una red compleja libre de escala, que pretende dar respuesta a la pregunta de investigación: ¿Cuáles son las dinámicas de la población de un algoritmo evolutivo tradicional basado en redes complejas libres de escala?

En la red propuesta cada nodo representa una solución al problema de optimización; cada enlace (edge) representa la generación descendiente. La generación *i* ejecuta el algoritmo evolutivo para generar otros individuos en la red. Una repetición de *j* se completa cuando se ha generado un número dado de generaciones, de modo que en cada generación se promedian 50 valores ($j = 50$)

para obtener la última generación de *i*. Este procedimiento se repite con 100 valores (*j = 100*) y cada generación resultó de los mejores promedios de los 50 y 100 valores, los promedios son necesarios porque permiten obtener un mayor significado estadístico de los resultados.

Se compararon cinco algoritmos evolutivos: evolutivo tradicional (Evolutive); evolutivo tradicional guiado por redes de pequeño mundo (Evolutive-SW) y guiado por redes de pequeño mundo con elección del mejor vecino (Evolutive-SW Best) (Triana, Bucheli y García, 2020b); evolutivo tradicional guiado por redes libres de escala (Evolutive-SF) y guiado por redes libres de escala con elección del mejor vecino (Evolutive-SF Best). Los dos últimos algoritmos son los que se proponen en este artículo.

El método propuesto consistió en obtener el fitness más bajo de las 50 y 100 repeticiones para cada algoritmo, y luego compararlos entre sí con el fin de identificar la tasa de convergencia de cada algoritmo. La Tabla 1 muestra los parámetros utilizados en el experimento para todos los algoritmos.

Tabla 1. Parámetros del experimento.

Parámetro	Valor
Número de repeticiones (*j*)	50 y 100
Razón de mutación	0.1
Número de generaciones	50

Funciones de optimización

El experimento se llevó a cabo con cuatro funciones de optimización: Ackley, Beale, Camel y Sphere (Back, 1996). Estas funciones (Figura 1) tienen mínimos globales y en algunos casos mínimos locales, lo cual es muy útil para pruebas de algoritmos evolutivos. El modelo de las figuras mencionadas se explica a continuación.

Función Ackley
La función Ackley se caracteriza por tener un espacio bidimensional, con mínimos locales y un mínimo global en *f(0, ..., 0) = 0*. Fue propuesta por Ackley (1987) y está definida por:

$$f(x_0 \cdots x_n) = -20exp(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}) - exp(\frac{1}{n}\sum_{i=1}^{n} cos(2\pi x_i)) + 20 + e$$

Ecuación 1. Función Ackley

Donde $x_i \in [-32, 32]$ para todo $i = 1, 2$.

Función Beale
La función Beale tiene un mínimo global en *x, y = (3, 0.5)* y se define como:

$$f(x,y) = (1.5 - x + xy)^2 + (2.25 - x + xy^2)^2 + (2.625 - x + xy^3)^2$$

Ecuación 2. Función Beale

Donde $x, y \in [-4.5, 4.5]$.

Función Camel
La función Camel es un espacio bidimensional, tiene seis mínimos locales, dos de los cuales son globales. Esta función se define como:

$$f(x,y) = 2x^2 - 1.05x^4 + \frac{x^6}{6} + xy + y^2$$

Ecuación 3. Función Camel

Donde $x, y \in [-5, 5]$.

Función Sphere
Se define como:

$$f(x_0 ... x_n) = \sum_{i=1}^{n} x_i^2$$

Ecuación 4. Función Sphere

Algoritmos utilizados

Algoritmo evolutivo

El algoritmo evolutivo que se utilizó fue el DE propuesto por Storn (1996). Se diferencia de otros métodos evolutivos porque las operaciones de

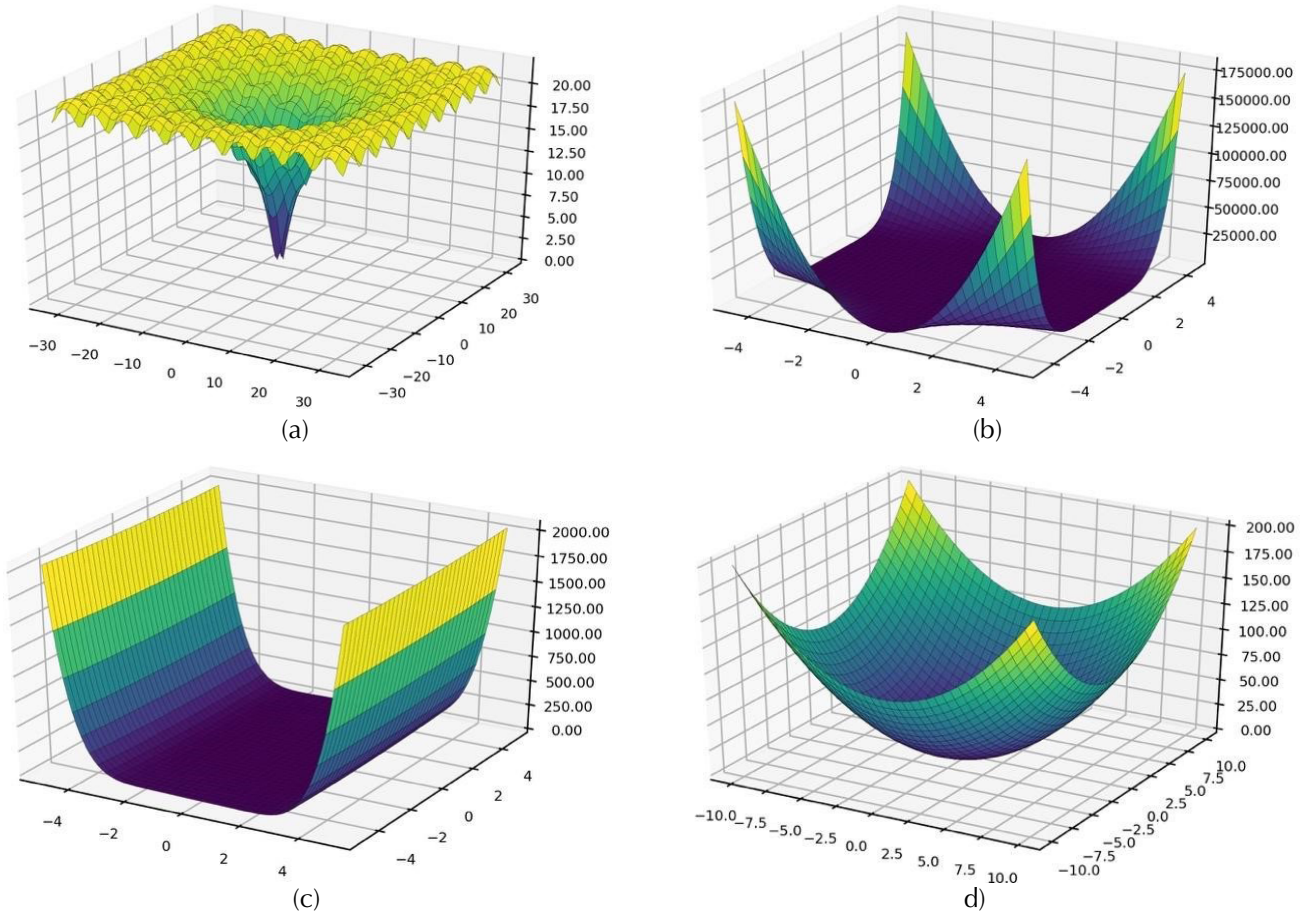


Figura 1. Visualización de funciones: (a) Función Ackley (b) Función Beale (c) Función Camel (d) Función Sphere

mutación y cruzamiento no son separables. Además, utiliza vectores para aplicar las operaciones, lo que hace que la convergencia sea más rápida. Comienza con una población aleatoria de individuos (NP), cada individuo se denota como X_{iG} donde *i = 0,1,2 ... NP-1* será un vector de características en un espacio *D* con una distribución normal. Posteriormente se seleccionan aleatoriamente tres vectores para crear un vector *u*, que pertenece a la siguiente generación, de acuerdo con la ecuación:

$$v_i = x_{1G} + F(x_{2G} - x_{3G})$$

Ecuación 5. Algoritmo evolutivo

Con *1G, 2G, R3G* ∈ *[0, NP-1]* entero, elegidos al azar y mutuamente diferentes y *F > 0F* es un factor constante que controla la amplificación de la variación diferencial. El vector *u* que pertenece a la siguiente generación viene dado por la Figura 7. El componente *j* del vector *u* será igual al componente *j* del vector *v* si un número aleatorio generado por cada componente entre 0 y 1 es menor o igual a *CR*. Si el número aleatorio es mayor que *CR*, el componente *j* de *u* es igual al componente *j* del vector X_{1G}, con *CR* ∈ *[0,1]*. Además, debe asegurarse que al menos uno de los componentes *j* seleccionados al azar del vector *u* sea igual al componente *j* del vector *v*. Esto se realiza de esa manera con el objetivo de que el vector *u* no sea idéntico al vector X_{1G}, en caso de que el

número aleatorio sea mayor que *CR*, para todos los componentes.

$$u_{ij,G+1} = \begin{cases} v_{ij} & \text{si} Random(0,1) <= CR \text{ or } j = j_{rand} \\ X_{j,1G} & \text{si} Random(0,1) > CR \text{ and } j \neq j_{rand} \end{cases}$$

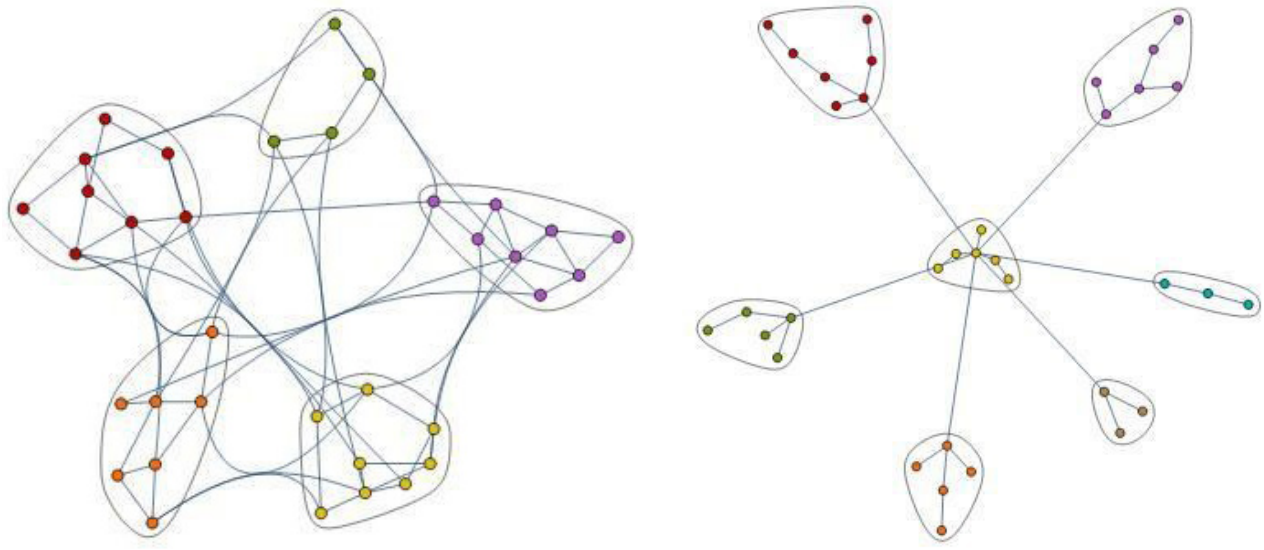
Ecuación 6. Especificación del método de reemplazo

Proponemos una variación a este método, debido a que no se crearán nuevos individuos en cada generación, sino que el padre será reemplazado por el hijo con mejor fitness.

Evolutivo-SW
Este método sigue el mismo algoritmo que el evolutivo propuesto, pero esta vez los individuos creados inicialmente son nodos dentro de una red compleja de pequeño mundo, cuyas propiedades se estudiaron primero en Milgram (1967). Cada uno de los nodos o individuos está relacionado con otro, pero esta vez la selección se limitará a los individuos vecinos. En la Figura 2 (a) se puede visualizar una red de pequeño mundo que se caracteriza porque la mayoría de sus nodos se pueden encontrar dentro de una distancia relativamente corta, aunque no sean vecinos. El modelo utilizado para generar esta red fue propuesto en Watts y Strogatz (1998).

Evolutivo-SW Best
Sigue el mismo algoritmo que se muestra en Evolutivo-SW, con la diferencia de que ahora los individuos se eligen al azar, favoreciendo aquellos con la mejor puntuación en la generación correspondiente.

Evolutivo-SF Barabási
Este método sigue el mismo algoritmo evolutivo, pero esta vez los individuos creados inicialmente son nodos dentro de una red compleja libre de escala. Cada uno de los nodos o individuos está relacionado con otro, pero esta vez la selección no será aleatoria, sino que se limitará a los individuos vecinos. En la Figura 2 (b) se muestra una visualización de una red libre de escala. Este tipo de red fue estudiada por (Barabási y Albert, 1999) y tiene una gran cantidad de nodos con pocas conexiones y pocos nodos con muchas conexiones, que se denominan "hubs". Además, tiene una conexión preferencial entre sus nodos, generando una distribución de calificaciones que mejora la tasa de convergencia y el rendimiento.



(a) (b)

Figura 2. Visualización 2D de las redes complejas de pequeño mundo y libre de escala

Evolutivo-SF Barabási Best
Sigue el mismo algoritmo que se muestra en Evolutivo-SF Barabási, con la diferencia que ahora los individuos se eligen al azar, favoreciendo aquellos con la mejor puntuación en la generación dada.

Generación de individuos

El proceso comienza creando una población de individuos con la estrategia evolutiva. Luego se crea una red compleja libre de escala con el mismo número de nodos que el tamaño de la población. Se realiza una asociación pseudoaleatoria de un individuo de la población con un nodo de red para iniciar la iteración del algoritmo, donde cada individuo de la población selecciona un nodo vecino asociado para la reproducción. De esta manera, la reproducción del individuo está guiada por la red compleja libre de escala. Si el individuo recién generado tiene un mejor desempeño que el padre seleccionado, se realiza un reemplazo; así se mantiene el tamaño de la población. Este principio permite uniformidad y garantiza que la red tenga la misma estructura, preservando las propiedades de la red compleja libre de escala.

Recableado

Para agregar dinamismo a la red y evitar una convergencia prematura, ya que su topología no cambia de una generación a la siguiente, en Triana, Bucheli y García (2020b) se propone realizar un recableado, que consiste en elegir aleatoriamente dos pares de nodos *a-b* y *c-d* conectados entre ellos. Las conexiones entre estos nodos se redefinen aleatoriamente, por ejemplo, *a-c* y *b-d*. Si alguna de estas conexiones ya existe, el proceso se cancela y se intenta con otro par. Este método asegura que la distribución de grados de la red no cambie para que no se pierdan las propiedades de cada uno de los tipos utilizados.

Resultados

Los experimentos se llevaron a cabo con el fin de evaluar el algoritmo evolutivo para los problemas de optimización (problemas de minimización): funciones de Ackley, Beale, Camel y Sphere. Se estudiaron los algoritmos evolutivos identificando la mejor solución de los problemas de optimización según los criterios de rendimiento y calidad.

```
Poblacion ← población generada pseudoaleatoriamente de tamaño n
Redes complejas sin escala ← Cada nodo asociado a un individuo único de la población
while detenerCriterioNoAlcanzado do
    for i ← 1 to n do
        individuoA ← i individuo de población
        for i ← 1 to k do
            vecinosA ← conjunto de todo individuosA vecinos
            if longitud(vecinosA) > 0 then:
                individuoB ← elige una persona de vecinosA por un criterio dado
                EtapaReproduccion con individuoA y individuoB generar un nuevo individuoC
                EtapaMutacion con individuoC
                if (individuoC.fitness) > (individuoA.fitness) then:
                    ReemplaceindividuoA con individuoC en poblacion
                end if
            end if
        end for
    end for
end while
```

Figura 3. Pseudocódigo del método propuesto para la generación de individuos

El algoritmo evolutivo tradicional se comparó con los algoritmos evolutivos: pequeño mundo (SW), pequeño mundo best (SW Best), redes libres de escala (Barabási) y redes libres de escala best (Barabási Best). En este trabajo proponemos los algoritmos evolutivos libres de escala (Barabási) y libres de escala best (Barabási Best). Realizamos el experimento con los algoritmos propuestos para una población con 50 individuos, se generaron diferentes iteraciones (50, 100, 500, 100 y 5000) con el fin de observar y analizar el rendimiento de los algoritmos, su dinámica y el resultado mínimo.

Para obtener los resultados más confiables, utilizamos los mejores promedios de solución en cada iteración. De esta forma, traemos la mejor solución y la evolución del modelo a lo largo del tiempo. Los resultados solo incluyen los mejores promedios para 50 y 100 iteraciones (ver las Tablas 2, 3, 4, 5 y 6). En las siguientes líneas se presentan los resultados obtenidos para las funciones Ackley, Beale, Camel y Sphere.

Función Ackley

La Tabla 2 presenta los cinco mejores promedios encontrados en orden descendente para 50 y 100 iteraciones. El algoritmo con mejor desempeño en las 50 iteraciones es Barabási, con un promedio final de 0.059073. Pero en 100 iteraciones el mejor resultado promedio es Barabási Best con 0.058757. Sin embargo, para ambos casos, los valores más altos se obtienen mediante el algoritmo evolutivo tradicional, con 0.132497 para 50 iteraciones y 0.126067 en 100 iteraciones.

La Figura 4 muestra una gráfica logarítmica de la minimización de la función de Ackley. Las figuras presentan resultados para 50 iteraciones (a) y 100 iteraciones (b), respectivamente. La figura (a) muestra que el algoritmo Barabási Best (línea azul) obtiene las mejores soluciones. Desde las primeras iteraciones Barabási Best muestra los resultados promedio más bajos, pero en las últimas iteraciones es superado por el algoritmo Barabási (línea amarilla). La figura (b) muestra el algoritmo Barabási Best con los mejores promedios en todas las iteraciones. Sin embargo, en ambas figuras observamos que el algoritmo evolutivo tradicional (línea verde) tiene los valores más altos en todas las iteraciones.

Función Beale

La Tabla 3 presenta los cinco mejores promedios dispuestos en orden descendente para 50 y 100 iteraciones. El algoritmo con mejor desempeño en

Tabla 2. Mejores promedios de la función Ackley para 50 y 100 interacciones

#	Iteraciones	Evolutivo	SW	SW Best	Barabási	Barabási Best
1	50	0.148225	0.072246	0.074484	0.059883	0.072218
2	50	0.148225	0.071563	0.072697	0.059883	0.066759
3	50	0.141862	0.066982	0.069564	0.059077	0.065971
4	50	0.135294	0.066924	0.069088	0.059073	0.064897
5	50	0.132497	0.065348	0.068860	**0.059073**	0.064897
1	100	0.143142	0.075657	0.090147	0.081467	0.066868
2	100	0.140999	0.074292	0.088918	0.079852	0.062414
3	100	0.138264	0.069637	0.086906	0.077785	0.059840
4	100	0.129330	0.068320	0.086450	0.073786	0.059308
5	100	0.126067	0.068156	0.081161	0.071457	**0.058757**

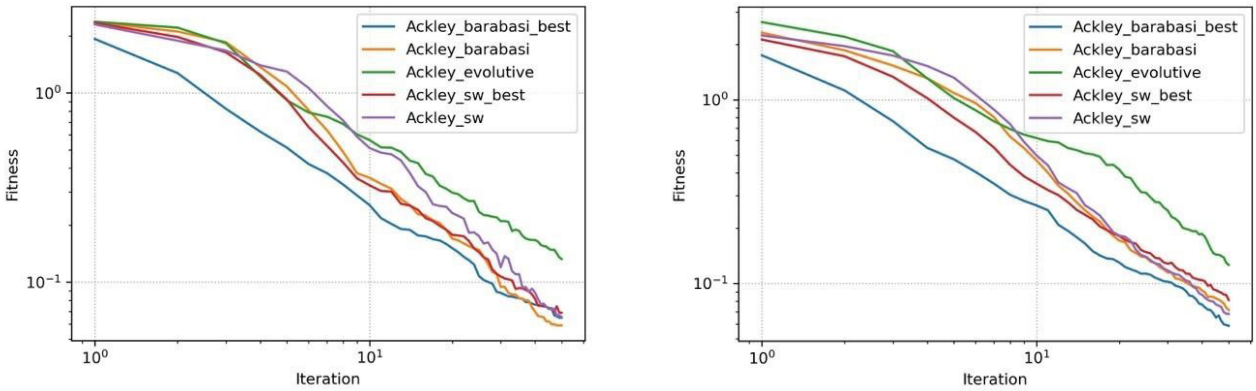


Figura 4. Experimentos implementados: (a) Función Ackley con 50 iteraciones (b) Función Ackley completa

ambos casos es Barabási, con una solución mínima de 0.010118 (50 iteraciones) y 0.012057 (100 iteraciones). Además, al igual que la función Ackley, el algoritmo evolutivo tradicional tiene las soluciones más altas: 0.035203 (50 iteraciones) y 0.058131 (100 iteraciones).

La Figura 5 muestra una gráfica logarítmica de la minimización de la función de Beale. Las figuras presentan resultados para 50 iteraciones (a) y 100 iteraciones (b), respectivamente. Las figuras (a) y (b) muestran el algoritmo Barabási (línea amarilla) con los mejores promedios en todas las iteraciones. El algoritmo evolutivo tradicional (línea verde) tiene los valores más altos en todas las iteraciones en ambas figuras y confirma que el enfoque novedoso tiene mejores resultados que el algoritmo evolutivo tradicional.

Función Camel

La Tabla 4 presenta los mejores resultados promedio en orden descendente para 50 y 100 iteraciones respectivamente. El algoritmo con mejor desempeño en las 50 iteraciones es Barabási 0.000043. Sin embargo, cuando llega a 100 iteraciones el mejor resultado promedio es Barabási Best 0.000032. Por otro lado, para ambos casos, los valores más altos se calculan mediante el algoritmo evolutivo tradicional con 0,000120 (50 iteraciones) y 0,000109 (100 iteraciones).

La Figura 6 muestra una gráfica logarítmica de la minimización de la función Camel. Las figuras presentan resultados para 50 iteraciones (a) y 100 iteraciones (b), respectivamente. La figura (a) muestra que el algoritmo de Barabási (línea amarilla) obtiene las mejores soluciones. La figura (b) muestra el algoritmo Barabási Best (línea azul) con los mejores promedios en todas las iteraciones. La función Camel presenta un comportamiento similar con las funciones Ackley y Beale, confirmando que el mecanismo guiado por las redes Barabási son mejores que el algoritmo evolutivo.

Función Sphere

La Tabla 5 presenta los mejores resultados promedio en una forma similar a las tablas anteriores. El algoritmo con mejor desempeño en ambos casos es Barabási Best, con un valor mínimo de 0.000014 para 50 y 100 iteraciones. El algoritmo evolutivo tradicional tiene las soluciones más altas: 0.000096 (50 iteraciones) y 0.000064 (100 iteraciones).

La Figura 7 muestra una gráfica logarítmica de la minimización de la función de Beale. Las figuras presentan resultados para 50 iteraciones (a) y 100 iteraciones (b), respectivamente. Las figuras (a) y (b) muestran el algoritmo Barabási Best (línea azul) con las mejores puntuaciones. Por el contrario, el algoritmo evolutivo tradicional (línea verde) tiene los valores más altos en todas las iteraciones.

Tabla 3. Mejores promedios de la función Beale para 50 y 100 interacciones

#	Iteraciones	Evolutivo	SW	SW Best	Barabási	Barabási Best
1	50	0.037083	0.019481	0.012572	0.010802	0.027503
2	50	0.036001	0.019165	0.012572	0.010586	0.026668
3	50	0.035634	0.019032	0.012461	0.010364	0.026603
4	50	0.035364	0.018765	0.012454	0.010290	0.026247
5	50	0.035203	0.018724	0.012399	**0.010118**	0.026214
1	100	0.059788	0.014587	0.025757	0.012734	0.056356
2	100	0.059639	0.014418	0.025596	0.012710	0.056174
3	100	0.059257	0.014245	0.025404	0.012270	0.055723
4	100	0.058511	0.014184	0.025396	0.012158	0.055711
5	100	0.058131	0.013654	0.025243	**0.012057**	0.055055

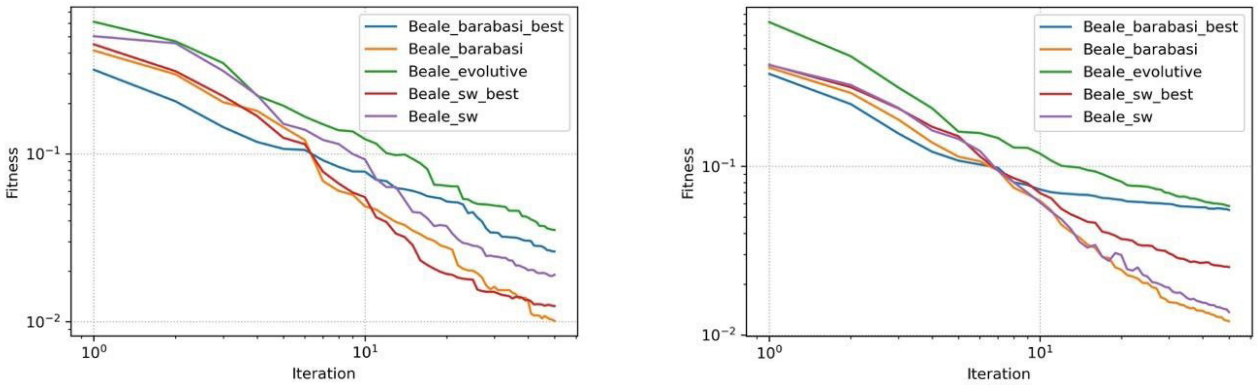
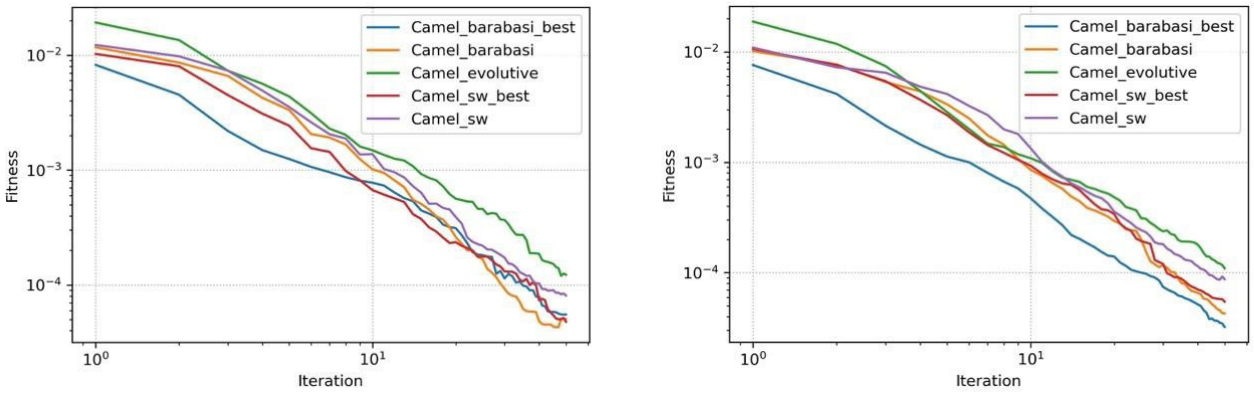


Figura 5. Experimentos implementados: (a) Función Beale con 50 iteraciones (b) Función Beale completa

Tabla 4. Mejores promedios de la función Camel para 50 y 100 interacciones

#	Iteraciones	Evolutivo	SW	SW Best	Barabási	Barabási Best
1	50	0.000143	0.000085	0.000052	0.000051	0.000058
2	50	0.000140	0.000085	0.000051	0.000051	0.000056
3	50	0.000125	0.000084	0.000050	0.000051	0.000055
4	50	0.000123	0.000083	0.000050	0.000043	0.000055
5	50	0.000120	0.000081	0.000048	**0.000043**	0.000055
1	100	0.000129	0.000092	0.000059	0.000050	0.000037
2	100	0.000123	0.000091	0.000058	0.000047	0.000037
3	100	0.000121	0.000087	0.000058	0.000046	0.000035
4	100	0.000117	0.000087	0.000058	0.000043	0.000035
5	100	0.000109	0.000086	0.000055	0.000043	**0.000032**

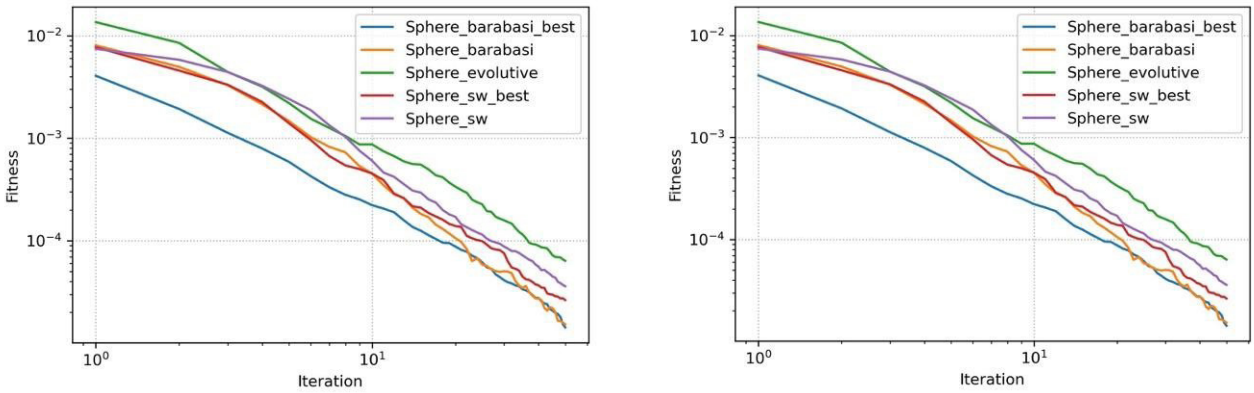


(a) (b)

Figura 6. Experimentos implementados: (a) Función Camel con 50 iteraciones (b) Función Camel completa

Tabla 5. Mejores promedios de la función Sphere para 50 y 100 interacciones

#	Iteraciones	Evolutivo	SW	SW Best	Barabási	Barabási Best
1	50	0.000102	0.000027	0.000043	0.000032	0.000017
2	50	0.000101	0.000027	0.000038	0.000032	0.000015
3	50	0.000099	0.000027	0.000036	0.000032	0.000015
4	50	0.000099	0.000026	0.000034	0.000029	0.000015
5	50	0.000096	0.000025	0.000034	0.000029	**0.000014**
1	100	0.000070	0.000044	0.000029	0.000020	0.000020
2	100	0.000069	0.000040	0.000028	0.000017	0.000019
3	100	0.000069	0.000039	0.000027	0.000016	0.000018
4	100	0.000066	0.000037	0.000027	0.000016	0.000015
5	100	0.000064	0.000036	0.000026	0.000015	**0.000014**



(a) (b)

Figura 7. Experimentos implementados: (a) Función Sphere con 50 iteraciones (b) Función Sphere completa

Los resultados se encuentran resumidos en la Tabla 6, se observa que los mejores promedios se obtienen con los algoritmos propuestos, es decir, los resultados alcanzados en las redes Barabási y Barabási Best superaron en todas las pruebas a los resultados del algoritmo evolutivo tradicional. También se puede observar que los resultados son consistentes en las funciones de optimización: Ackley, Beale, Camel y Sphere. Los problemas de optimización de la función de prueba se minimizan con el enfoque propuesto, en el que Barabási Best fue el mejor algoritmo para las funciones Ackley, Camel y Sphere con los siguientes promedios: 0.058757, 0.000032 y 0.000014. No obstante, el mejor rendimiento del algoritmo Barabási lo tiene la función Beale con 0.012057.

Conclusiones

La computación evolutiva tradicional es un algoritmo iterativo que resuelve problemas de optimización y los individuos se combinan para obtener una mejor solución. Estos individuos generan nuevas soluciones que permiten resolver el problema de manera óptima. Según Amara *et al*. (2006), las redes de pequeño mundo han contribuido a los problemas de optimización al usarlas como mecanismo de control en combinaciones de individuos. Las distancias entre los vértices tienden a ser más cortas que las redes aleatorias, y todos los elementos fuertemente relacionados entre sí permiten obtener el valor óptimo en menos tiempo. En Triana, Bucheli y García (2020b), se utilizó el algoritmo evolutivo tradicional y una red de pequeño mundo para comparar cuatro funciones de optimización: Ackley, Beale, Camel y Sphere. Los resultados muestran que la generación de individuos (soluciones) mejoró con la adición de la red de pequeño mundo, porque el modelo desarrollado minimizó los promedios en menos iteraciones que el algoritmo tradicional.

El documento de Triana, Bucheli y García (2020b) propone dos variantes de la selección del vecino a los encontrados en la literatura. El primero, llamado pequeño mundo, generó una selección aleatoria del vecino para mejorar la combinación de soluciones. El segundo, denominado pequeño mundo best, consistió en seleccionar al vecino mediante el método de la ruleta para favorecer el mejor rendimiento (fitness). En los resultados se observó que las dos variantes generaron mejores promedios en comparación con la estrategia evolutiva tradicional para los parámetros: tamaño de la población (100 individuos), tasa de mutación (10 %) y criterio de parada (50 iteraciones).

Este trabajo se desarrolló a partir del trabajo propuesto por Triana, Bucheli y García (2020b). En comparación con las literaturas encontradas, la contribución de este trabajo se basa en agregar una red libre de escala o Barabási al proceso de combinación de soluciones (Barabási y Albert, 1999). Los problemas de optimización que se estudiaron fueron: Ackley, Beale, Camel y Sphere. Los resultados de los algoritmos propuestos se compararon con la estrategia de computación evolutiva tradicional y el algoritmo evolutivo redes de pequeño mundo. En el experimento se utilizaron los mismos parámetros que en Triana, Bucheli y García (2020b), es decir: la misma selección de individuos, recableado, criterios de parada, tamaño de población y tasa de mutación. Sin embargo, los algoritmos se

Tabla 6. Mejores promedios para las funciones Ackley, Beale, Camel y Sphere con 100 iteraciones

Función	Iteraciones	Evolutivo	SW	SW Best	Barabási	Barabási Best
Ackley	100	0.126067	0.068156	0.081161	0.071457	**0.058757**
Beale	100	0.058131	0.013654	0.025243	**0.012057**	0.055055
Camel	100	0.000109	0.000086	0.000055	0.000043	**0.000032**
Sphere	100	0.000064	0.000036	0.000026	0.000015	**0.000014**

ejecutaron para 50 y 100 iteraciones (criterios de parada). En todas las pruebas realizadas se observó que los algoritmos propuestos en este trabajo (red Barabási) presentan mejores promedios en comparación con la computación evolutiva tradicional, pequeño mundo y pequeño mundo best, para las 50 y 100 iteraciones. Este resultado confirma el trabajo desarrollado por Triana, Bucheli y García (2020b), encontrando que, para 100 iteraciones, los resultados se mantienen. De esta forma se pueden corroboran los resultados y se mejora la propuesta realizada en Triana, Bucheli y García (2020) agregando una red Barabási y obteniendo mejores promedios en el proceso de optimización. El trabajo propuesto utiliza un mecanismo de conexión preferencial entre sus nodos, generando una distribución de grados de ley de potencia de los nodos (Adamic *et al*., 2001), asociándose con individuos. Además, guía la combinación de individuos (soluciones), mejorando la tasa de convergencia y el rendimiento del algoritmo evolutivo en general.

Se observó que los algoritmos propuestos presentan mejores resultados que las funciones de optimización (minimización) Ackley, Beale, Camel y Sphere. Así, para la función Sphere con pequeño mundo se obtiene 0,000025, mientras que Barabási Best mejoró los valores en 0,000006, pasando de 0,000020 a 0,000014. De los resultados observados en la función de Ackley, se obtiene 0.065348 en pequeño mundo, mientras que Barabási Best mejoró los valores en 0.013461, pasando de 0.072218 a 0.058757. En la función Camel se obtiene 0.000081 con pequeño mundo. Sin embargo, Barabási Best mejoró los promedios en 0,000026, pasando de 0,000058 a 0,000032. Estos resultados nos permiten concluir que el rendimiento de los algoritmos evolutivos mejora con alguna versión de la red libre de escala (Barabási o Barabási Best). En general, la red Barabási como mecanismo de control de combinación de soluciones es mejor que el algoritmo evolutivo tradicional o el algoritmo evolutivo guiado por redes de pequeño mundo.

Como trabajo futuro se propone implementar algoritmos evolutivos con un mecanismo de control guiado por redes Barabási al problema de optimización combinatoria del vendedor viajero (TSP) y al problema de la mochila (KP). El primer problema busca identificar la ruta óptima (la distancia más corta entre ciudades), visitando cada ciudad exactamente una vez, pero regresando a la ciudad de origen. En el otro problema se modelan diferentes soluciones en el llenado de la mochila, maximizando el total de los objetos sin exceder el peso definido. Los dos problemas se utilizan ampliamente en el mundo real y la contribución de incluir una red libre de escala permitiría encontrar la solución más óptima en menos tiempo.

Reconocimientos

Este trabajo ha sido apoyado por el Departamento de Ingeniería y Ciencias de la Computación de la Universidad del Valle.

Referencias

Ackley, D. H. (1987). *A Connectionist Machine for Genetic Hillclimbing* (Vol. 28). Springer US. https://doi.org/10.1007/978-1-4613-1997-9

Adamic, L. A., Lukose, R. M., Puniyani, A. R., Huberman, B. A. (2001). Search in power-law networks. *Physical Review E*, *64*(4), e46315. https://doi.org/10.1103/PhysRevE.64.046135

Amara, L. A. N., Scala, A., Barthelemy, M., Stanley, H. E. (2006). Classes of small-world networks. In *The Structure and Dynamics of Networks* (pp. 207-210). Princeton University Press. https://www.degruyter.com/document/doi/10.1515/9781400841356.207/html

Back, T. (1996). *Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, Genetic Algorithms*. Oxford University Press

Barabási, A.-L., Albert, R. (1999). Emergence of scaling in random networks. *Science*, *286*(5439), 509-512. https://doi.org/10.1126/science.286.5439.509

Bremermann, H. J. (1958). *The Evolution of Intelligence: The Nervous System as a Model of its Environment*. University of Washington, Department of Mathematics

Fogel, D. B. (2005). Theoretical and empirical properties of evolutionary computation. In *Evolutionary Computation: Toward a New Philosophy of Machine Intelligence* (pp.105-181). John Wiley & Sons, Inc. https://doi.org/10.1002/0471749214

Friedman, G. J. (1956). *Selective Feedback Computers for Engineering Synthesis and Nervous System Analogy*. *University of California at Los Angeles*

Kadavy, T., Pluhacek, M., Viktorin, A., Senkerik, R. (2017). Firework algorithm dynamics simulated and analyzed with the aid of complex network. En *ECMS 2017 Proceedings*, 313-318. https://doi.org/10.7148/2017-0313

Metlicka, M., Davendra, D. (2017). Complex network analysis of firefly algorithm population dynamics. En *IEEE Symposium Series on Computational Intelligence (SSCI)*, 1-8. https://doi.org/10.1109/SSCI.2017.8285371

Michalewicz, Z. (1996). Heuristic methods for evolutionary computation techniques. *Journal of Heuristics*, *1*(2), 177-206. https://doi.org/10.1007/BF00127077

Milgram, S. (1967). The small world problem. *Psychology Today*, *2*, 60-67

Pizzuti, C. (2018). Evolutionary computation for community detection in networks: A Review. *IEEE Transactions on Evolutionary Computation*, *22*(3), 464-483. https://doi.org/10.1109/TEVC.2017.2737600

Plagianakos, V. P., Tasoulis, D. K., Vrahatis, M. N. (2008). A review of major application areas of differential evolution. In U. K. Chakraborty (Ed.), *Advances in Differential Evolution* (pp. 197-238). Springer. https://doi.org/10.1007/978-3-540-68830-3_8

Sheng, L., Guang, X., Chen, F., Wang, H., Gao, K. (2016). A review on complex network dynamics in evolutionary algorithm. En *IEEE Trustcom/BigDataSE/ISPA*, 2221-2226. https://doi.org/10.1109/TrustCom.2016.0342

Storn, R. (1996). On the usage of differential evolution for function optimization. En *Proceedings of North American Fuzzy Information Processing*, 519-523. https://doi.org/10.1109/NAFIPS.1996.534789

Triana, J., Bucheli, V., García, A. (2020a). Traveling salesman problem solving using evolutionary algorithms guided by complex networks. *International Journal of Artificial Intelligence*, *18*(2), 101-112

Triana Madrid, J., Bucheli Guerrero, V., García Baños, Á. (2020b). Sistema de control para computación evolutiva basado en redes complejas. *Investigación e Innovación en Ingenierías*, *8*(2), 169-183

Wang, Z., Sobey, A. (2020). A comparative review between Genetic Algorithm use in composite optimisation and the state-of-the-art in evolutionary computation. *Composite Structures*, *233*, e111739. https://doi.org/10.1016/j.compstruct.2019.111739

Watts, D. J., Strogatz, S. H. (1998). Collective dynamics of 'small-world' networks. *Nature*, *393*(6684), 440-442. https://doi.org/10.1038/30918

Yinghui, Y., Jianhua, L., Di, S., Mingli, N., Qiong, C. (2018). Evolutionary dynamics analysis of complex network with fusion nodes and overlap edges. *Journal of Systems Engineering and Electronics*, *29*(3), 549-559. https://doi.org/10.21629/JSEE.2018.03.12

Zelinka, I. (2011). Investigation on relationship between complex networks and evolutionary algorithms dynamics. *AIP Conference Proceedings*, *1389*(1), 1011-1014. https://doi.org/10.1063/1.3637781

Zelinka, I., Davendra, D., Lampinen, J., Senkerik, R., Pluhacek, M. (2014). Evolutionary algorithms dynamics and its hidden complex network structures. En *IEEE Congress on Evolutionary Computation*, 3246-3251. https://doi.org/10.1109/CEC.2014.6900441

Zelinka, I., Davendra, D., Roman, S., Roman, J. (2011). Do evolutionary algorithm dynamics create complex network structures? *Complex Systems*, *20*(2), 127-140. https://doi.org/10.25088/ComplexSystems.20.2.127

Zelinka, I., Davendra, D., Snasel, V., Jasek, R., Senkerik, R., Oplatkova, Z. (2010). Preliminary investigation on relations between complex networks and evolutionary algorithms dynamics. En *International Conference on Computer Information Systems and Industrial Management Applications*, 148-153. https://doi.org/10.1109/CISIM.2010.5643674

Zelinka, I., Tomaszek, L., Snasel, V. (2015). On evaluation of evolutionary networks using new temporal centralities algorithm. En *International Conference on Intelligent Networking and Collaborative Systems*, 334-339. https://doi.org/10.1109/INCoS.2015.95