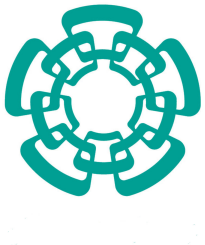


Centro de Investigación y de Estudios Avanzados
del Instituto Politécnico Nacional

Unidad Cinvestav Tamaulipas

Un Algoritmo Cultural para el Problema de la Ubicación de Estaciones de Carga de Vehículos Eléctricos

Tesis que presenta:

Norma Lucero Cuautle Rivera

Para obtener el grado de:

**Maestro en Ciencias
en Ingeniería y Tecnologías
Computacionales**

Director de la Tesis:
Dr. Ricardo Landa Becerra

Cd. Victoria, Tamaulipas, México. Noviembre, 2017

La tesis presentada por Norma Lucero Cuautle Rivera fue aprobada por:

_____

_____
Dr. Eduardo Arturo Rodríguez Tello

_____
Dr. Gregorio Toscano-Pulido

_____
Dr. Ricardo Landa Becerra, Director

Cd. Victoria, Tamaulipas, México., 15 de Noviembre de 2017

A mi familia

Agradecimientos

- Agradezco a mi amado esposo Carlos Hernández por su constante apoyo y aliento en cada uno de los momentos en que lo necesite. A pesar de sus tantas actividades, siempre tuve sus palabras de aliento y comprensión, las cuales fueron esenciales para cursar mis estudios de posgrado.
- Doy gracias a mi familia por su apoyo en la distancia, a mi mai Catalina, a mis hermanas Adriana y Marisol y a mi sobrinita Paulita.
- Agradezco a mi director de tesis Dr. Ricardo Landa y a mis revisores Dr. Eduardo Arturo Rodríguez Tello y Dr. Gregorio Toscano Pulido por sus consejos y tiempo dedicado para el desarrollo de la tesis. Así también, a mis compañeros de la maestría por su camaradería, aprecio y los buenos recuerdos de todos y cada uno de ellos.
- Agradezco al personal administrativo del CINVESTAV-Tamaulipas por su ayuda durante mi estancia.
- También a CONACyT por el apoyo económico y CINVESTAV-Tamaulipas por la oportunidad de cursar mis estudios de posgrado.

Índice General

Índice de Figuras

Índice de Tablas

Índice de Algoritmos

Publicaciones

Resumen

Un Algoritmo Cultural para el Problema de la Ubicación de Estaciones de Carga de Vehículos Eléctricos

por

Norma Lucero Cuautle Rivera
Unidad Cinvestav Tamaulipas
Centro de Investigación y de Estudios Avanzados del Instituto Politécnico Nacional, 2017
Dr. Ricardo Landa Becerra, Director

Al hablar del sistema de transporte actual y las fuentes de energía renovables, los vehículos eléctricos son los mayores exponentes como alternativa al uso de combustibles fósiles. La incorporación al sistema de trasporte existente de dichos vehículos con su limitado rango de movimiento, requiere de una planeación que enfrente los retos de contar con suficientes estaciones de carga y el fácil acceso a las mismas, dada la autonomía de los vehículos eléctricos. En este trabajo se propone un algoritmo cultural para resolver el problema de la ubicación de las estaciones de carga de los vehículos eléctricos minimizando el costo de construcción. Se presenta el diseño de cada uno de los componentes que integran al algoritmo propuesto. Se diseñaron cuatro fuentes de conocimiento, a saber: de dominio con costos, de dominio con puntos de articulación, situacional y normativo; teniendo un mayor número de casos de éxitos (frecuencia en que cada una de las fuentes de conocimiento mejoran la aptitud de los individuos mutados en cada generación) tanto la fuente de conocimiento de dominio con costos, como la fuente de conocimiento normativo. De ellas, en la primera se empleó conocimiento del dominio del problema en cuanto a los costos de construcción de las estaciones de carga, mientras que en la segunda se seleccionaron para la construcción aquellos nodos (posibles ubicaciones de estaciones de carga) que siguieran la tendencia del conjunto de individuos aceptados.

Para evaluar su desempeño se compararon los resultados obtenidos primeramente con un algoritmo genético simple y posteriormente con un algoritmo voraz del estado del arte. Los resultados indican que el uso de un algoritmo cultural en la solución del problema de la ubicación de las estaciones de carga de vehículos eléctricos provee soluciones competitivas.

Abstract

A Cultural Algorithm for the Electric Vehicle Charging Station Placement Problem

by

Norma Lucero Cuautle Rivera

Cinvestav Tamaulipas

Center for Research and Advanced Studies of the National Polytechnic Institute, 2017

Dr. Ricardo Landa Becerra, Advisor

When talking about the current transportation system and renewable energy sources, electric vehicles are the greatest exponents as an alternative to fossil-fuel vehicles. The incorporation to the existing transportation system of these vehicles with their limited autonomy, requires planning that faces the challenges of having sufficient charging stations and easy access to them. In this thesis we propose a cultural algorithm to solve the electric vehicle charging station placement problem, minimizing the cost of construction. We present the design of each of the components that integrate the proposed algorithm. Four knowledge sources were designed, namely: domain with costs, domain with points of articulation, situational and normative; having a higher success cases number (frecuency in which each of the knowledge sources improves the fitness of the individuals mutated in each generation) both, the source of domain knowledge with costs and the source of normative knowledge. Of them, the first one used knowledge of the problem domain in terms of the construction costs of the charging stations, while in the second one those nodes (potential charging station construction sites) that followed the trend of the set of accepted individuals were selected for the construction. In order to evaluate its performance it is compared first with a simple genetic algorithm and later with a greedy algorithm of the state of the art. The results indicate that the use of a cultural algorithm in the solution of the electric vehicle charging station placement problem provides competitive solutions.

Nomenclatura

AC	Algoritmo cultural
AG	Algoritmo genético
AV	Algoritmo voraz
EV	Electric vehicle
PSO	Particle swarm optimization
IP	Integer programming
GA	Genetic algorithm
G	El grafo no dirigido que modela la ciudad
N	Conjunto de posibles ubicaciones para la construcción de las estaciones de carga
E	Conjunto de conexiones que conectan pares de ubicaciones para la construcción
n	Tamaño de N
$d(i, j)$	Distancia de la ruta más corta desde el nodo i al nodo j
f_i	Capacidad de carga del nodo i
F_i	Requerimiento de demanda del nodo i
D	Distancia media transitable de los vehículos eléctricos completamente cargados
N'	Conjunto de nodos con estaciones de carga construidas
α	Factor de tolerancia que los conductores tienen de alejarse de su ubicación para cargar su vehículo eléctrico
$h_{i,j}$	Número de saltos de la ruta más corta de los nodos i al j en G
x_i	Variable booleana que representa si el nodo i es seleccionado o no para la construcción de una estación de carga
x	Vector de x_i, con $0 \leq i \leq n$
c_i	Costo de construcción del nodo i
$N_i^{\alpha D}$	Conjunto de nodos con distancia menor que αD del nodo i
$\hat{G}$	Grafo inducido de G
$\hat{N}$	Conjunto de nodos en $\hat{G}$
$\hat{E}$	Conjunto de aristas en $\hat{G}$
H	Subgrafo inducido de $\hat{G}$
0^i	Nodo fuente de flujo conectado al nodo i
x_0^i	Residuo del flujo que se mantuvo en 0^i
y_{jk}^i	Cantidad de flujo en el arista (j, k) originado de 0^i
$N(H)$	Conjunto de nodos asociados a H
C	Costo de construcción máximo
$\tilde{G}$	Grafo no dirigido para el problema de la cobertura de vértices
$\tilde{N}$	Conjunto de nodos en $\tilde{G}$
$\tilde{E}$	Conjunto de aristas en $\tilde{G}$

x^j	El j-ésimo individuo de la población
x^{best}	Mejor individuo encontrado hasta la generación anterior
$x^{newBest}$	Nuevo mejor individuo recién encontrado en la generación actual

1

Introducción

A continuación se plantean los antecedentes y la motivación del problema, tanto a nivel medioambiental como desde el punto de vista computacional, así como el planteamiento del problema, modelado, formulación y complejidad computacional. En este capítulo también se muestran la hipótesis propuesta y los objetivos de este trabajo de tesis.

1.1 Antecedentes

El impulso de la adopción del vehículo eléctrico (EV, por sus siglas en inglés) en ciertas ciudades ha tenido un gran impacto como alternativa al uso de energía fósil y para contar con diferentes beneficios, tanto ecológicos como económicos. Algunos países han optado por el uso del EV debido a la promoción de nuevas tecnologías a favor del medio ambiente.

Con el uso del EV, al no consumir combustibles fósiles, reduciendo con ello el nivel de las emisiones de dióxido de carbono, se modera el efecto contaminante que se tiene debido al uso de los vehículos de combustión. Lo anterior conduce a mejorar la calidad del aire, y a su vez repercute en beneficios para la salud.

También es importante tomar en cuenta que la utilización del EV está relacionada con las fuentes de energía renovable [47], esto en países donde el origen de la electricidad es renovable, lo que hace que su movilidad sea sostenible. Otro de los beneficios del uso de los EV es el ahorro, tanto en el costo de energía por kilómetro recorrido (con respecto al gasto con uso de gasolina) como en costos de mantenimiento.

Para la integración de los vehículos eléctricos en el sistema de transporte existente, es indispensable afrontar el reto de planificar las instalaciones de carga, tomando en cuenta la limitada autonomía del EV. Esto conlleva la necesidad de contar con suficientes estaciones de carga en una ciudad; así también, es importante considerar los costos elevados de construcción de dichas estaciones. Por tales razones, es necesario determinar las mejores ubicaciones para construir estaciones de carga en una ciudad, minimizando el costo de construcción de tal manera que se tenga una cobertura amplia y, de este modo, lograr la conveniencia de los conductores.

El problema de determinar las ubicaciones para la construcción de estaciones de carga de vehículos eléctricos (EVCSP, por sus siglas en inglés) es un problema que se ha planteado y formalizado recientemente, debido a la creciente popularidad del concepto de ciudades inteligentes [13]. El problema ha sido definido en la literatura considerando varios aspectos y se ha enfrentado empleando diversos métodos.

Los estudios del problema de EVCSP se pueden dividir en dos áreas: la económica y la relacionada con la red de energía. En este trabajo se consideran las mejores ubicaciones para el EVCSP tomando en cuenta el beneficio económico. Dentro de las funciones de costo empleadas en los trabajos relacionados tenemos: las de construcción, operación, carga, transportación, entre otras. Entre los métodos que han sido empleados para intentar resolver el EVCSP se encuentran principalmente:

algoritmos genéticos (GA, por sus siglas en inglés), optimización basada en cúmulo de partículas [6, 44] (PSO, por sus siglas en inglés) y programación entera [9] (IP, por sus siglas en inglés) [17].

1.2 Motivación

El problema de determinar las ubicaciones para la construcción de estaciones de carga de vehículos eléctricos es un problema de optimización combinatoria NP-completo, lo cual se demuestra en el trabajo de Lam et al. [22], donde para formular la versión de decisión del EVCSP construyen una reducción del problema de la cobertura de vértices (VCP, por sus siglas en inglés) para el EVCSP.

Lo anterior hace que el problema sea computacionalmente interesante para atacarse con metaheurísticas modernas. En este trabajo de tesis se propone un algoritmo cultural para afrontar el problema del EVCSP, motivados por el éxito que han tenido los algoritmos culturales resolviendo otros problemas combinatorios como los de asignación, para los cuales se han obtenido buenos resultados como en Liu and Lin [26], Ochoa Zezzatti et al. [32] y Soza et al. [46]. Hasta el momento no hay registro en la literatura de que se hayan aplicado al EVCSP.

1.3 Planteamiento del problema

Para que la movilidad eléctrica se acabe imponiendo en las ciudades actualmente dependientes de vehículos de combustión interna, se requiere mayor infraestructura de recarga de baterías de los EV. Entre mayor sea el número de estaciones de recarga y éstas se encuentren lo mejor distribuidas posible, será más fácil moverse con un EV en toda la ciudad. Sin embargo, es importante considerar los elevados costos de construcción de las estaciones de carga. Dadas las razones anteriores, es imprescindible determinar las mejores ubicaciones para la construcción de las estaciones de carga, minimizando así su costo de construcción.

En Lam et al. [22] mencionan las siguientes tres condiciones que se deben cumplir en la selección de las ubicaciones para la construcción de las estaciones de carga:

1. Garantizar que un EV que ha sido totalmente cargado en una ubicación, pueda recargarse en otra a cierta distancia y que los EV no estén confinados en una sola ubicación.

2. Se debe satisfacer la demanda de carga en una ubicación mediante la contribución de las capacidades de carga de todas las estaciones de carga que se encuentran a una distancia αD, donde α es la tolerancia que los conductores tienen de alejarse de su ubicación para cargar y D es la distancia de la autonomía del auto.

3. La red de estaciones de carga, donde cada una está separada de otra a lo mucho por una distancia D, se debe extender por toda la ciudad.

1.3.1 Modelado y requerimientos básicos

Para modelar una ciudad se emplea un grafo $G = (N, E)$, donde N denota el conjunto de las posibles ubicaciones para la construcción de las estaciones de carga, con $|N| = n$ y donde E representa las conexiones entre pares de ubicaciones. Sea $d : N \times N \to \mathbb{R}$ la función de distancia, donde $d(i, j)$ denota la distancia de la ruta más corta desde el nodo i hasta el nodo j atravesando las conexiones. Sea f_i la capacidad media del servicio de carga si una estación de carga se construye en la ubicación i; esta capacidad se relaciona con el tamaño de la estación y las condiciones de tráfico en los alrededores. Cada nodo i tiene un requerimiento de demanda F_i, el cual se refiere a su demanda local de carga media; F_i puede estimarse a partir del tamaño de la población y la tasa de penetración de vehículos eléctricos de esa ubicación.

Se define D como la distancia promedio que un vehículo eléctrico básico del mercado es capaz de recorrer cuando está cargado totalmente. Un subconjunto de nodos $N' \subseteq N$ es *accesible* por D si cumple las siguientes condiciones:

1) Para cada nodo $i \in N'$, existe un nodo $j \in N'$ tal que $d(i,j) \leq D$.

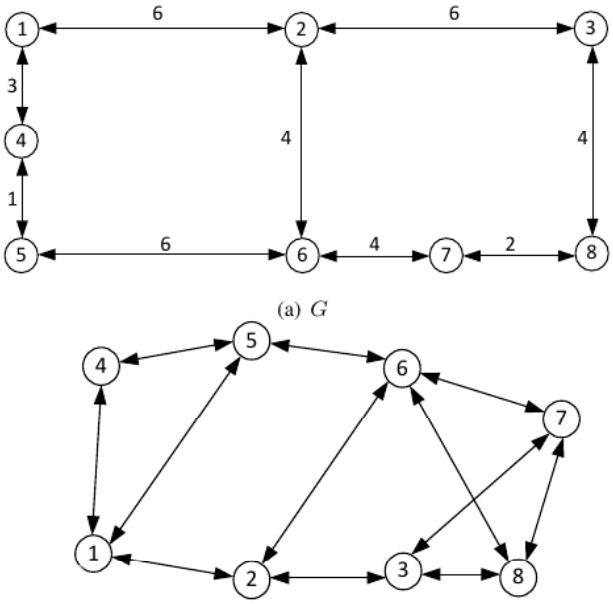
2) Para cada $i \in N$, la capacidad total, definida a partir de los nodos $j \in N'$ tal que $d(i,j) \leq \alpha D$ (donde $\alpha \in (0,1]$ corresponde al factor de tolerancia), es mayor o igual a F_i.

3) Para todo $i, j \in N'$, sea h_{ij} el número de saltos de la ruta más corta desde i a j en G. La distancia de la ruta $d(i,j)$ debe ser menor o igual a $h_{ij}D$.

1.3.2 Formulación

Se define x_i como una variable de decisión booleana que indica si el nodo i se elige para construir una estación de carga; a su vez, se define c_i como el costo de construcción de tal estación. Para minimizar el costo total se emplea la función objetivo $\sum_{i=1}^{n} c_i x_i$. Para cada i, se define $N_i^{\alpha D} = \{j \in N | d(i,j) \leq \alpha D\}$ representando el conjunto de nodos (incluyendo el mismo nodo i) con distancia menor o igual que αD, en relación con i. Habiendo definido tales términos podemos proceder a plantear las condiciones matemáticamente. La condición 2 se plantea como $\sum_{j \in N_i^{\alpha D}} f_j x_j \geq F_i \; \forall i \in N$. Como la condición 3 se mantiene para todo par de nodos, la condición 3 implica la condición 1. Para plantear la condición 3, primero se creó un grafo $\hat{G} = (N, \hat{E})$ donde $\hat{E} = \{(i,j) | i,j \in N, d(i,j) \leq D, i \neq j\}$. Para evaluar cada solución particular al problema, se consideran los nodos i en G con $x_i = 1$ (es decir, N') para constituir el correspondiente subgrafo inducido H de $\hat{G}$. En la Figura 1.1a se muestra un ejemplo del grafo G con 8 nodos, donde el número sobre cada arista indica la distancia entre los dos nodos que une. Con una distancia $D = 6$ se tiene el grafo $\hat{G}$ en la Figura 1.1b. La condición 3 es equivalente a tener H conexo; en otras palabras, H tiene un solo componente conexo.

En vez de inspeccionar el grafo original G, por facilidad se inspecciona $\hat{G}$ para formular el problema. Similar a [2], se adoptó un modelo de flujo de red para evaluar la condición 3. Se considera que hay algún flujo virtual de algunas fuentes hacia algunos destinos. Si las fuentes y los destinos no están conectados, el flujo de las fuentes no alcanza los destinos. Se supone que hay un nodo fuente



(a) G

(b) $\hat{G}$

Figura 1.1: Ejemplo de un grafo que modela un problema con $n = 8$ y $D = 6$

0^i ligado al nodo i y éste tiene n unidades de flujo disponibles para ser enviadas a lo largo de $\hat{G}$. Se permite a $0 \leq x_0^i \leq n$ ser el residuo de flujo no consumido por la red. Cada nodo j con $x_j = 1$ consumirá una unidad de flujo. Para cada arista $(i, k) \in \hat{E}$ se indica la cantidad de flujo en (j, k) originado de 0^i con la variable y_{jk}^i.

Por lo tanto, se puede garantizar que el flujo puede alcanzar estos nodos j con $x_j = 1$ desde el nodo i en $\hat{G}$ con lo siguiente:

$$x_0^i + y_{0i}^i = n, \tag{1.1}$$

$$0 \leq y_{jk}^i \leq nx_k, \forall (j, k) \in \hat{E} \cup \{0^i, i\}, \tag{1.2}$$

$$\sum_{j|(j,k)\in\hat{E}} y_{jk}^i = x_k + \sum_{l|(k,l)\in\hat{E}} y_{k,l}^i, \forall k \in \hat{N} \tag{1.3}$$

$$\sum_{j\in\hat{N}} x_j = y_{0i}^i, \tag{1.4}$$

$$0 \leq x_0^i \tag{1.5}$$

La Ecuación 1.1 indica que la cantidad total de flujo de salida y_{0i}^{i} de la fuente 0^i y el flujo retenido en x_0^i en 0^i es n, donde n es el número de nodos en G y éste es el límite superior del flujo posible para ser absorbido en la red. La Ecuación 1.2 dice que solamente los nodos destino puede recibir flujo de entrada. La Ecuación 1.3 describe que el flujo total de entrada a cualquier nodo destino es igual al flujo total de salida más la cantidad consumida por tal nodo. La Ecuación 1.4 explica que el flujo total obtenido de la fuente es igual al flujo total absorbido por los destinos. Finalmente, la Ecuación 1.5 es la restricción de no negatividad para el residuo restante en la fuente.

1.4 Complejidad computacional

A continuación se reproducen los teoremas y pruebas que aparecen en el trabajo de Lam et al. [22].

El EVCSP en su versión de decisión puede ser formulado de la siguiente manera: Sea $N(H)$ el conjunto de nodos asociados al subgrafo inducido H. Cada nodo i tiene una capacidad $f_i \in \mathbb{Z}^+$ y una demanda F_i; a su vez, el nodo i está asociado con el conjunto de nodos $N_i^{\alpha D}$. Dado un grafo no dirigido $\hat{G} = (\hat{N}, \hat{E})$, con un costo de construcción $c_i \in \mathbb{Z}^+, \forall i$, y un costo límite $C \in \mathbb{Z}^+$, ¿Existe un subgrafo inducido H de $\hat{G}$ tal que (i) Para cada $i \in N, \sum_{j \in N_i^{\alpha D} \cap N(H)} f_j \geq F_i$; (ii) H es conexo; y (iii) $\sum_{i \in N(H)} c_i \leq C$?

Teorema 1.4.1 *La versión de decisión del EVCSP es NP-completo.*

Se construyó una reducción del problema de la cobertura de vértices (VCP) para el EVCSP. En el grafo $\tilde{G} = (\tilde{N}, \tilde{E})$, una cobertura de vértices es un subconjunto de nodos $N' \subset \tilde{N}$ tal que cada arista $(i, j) \in \tilde{E}$ tienen cualquiera i o j, o ambos en $\tilde{N}$. Se asume que $\tilde{E} \neq \emptyset$. El VCP determina si existe una cubierta de vértices N' de $\tilde{G}$ con $|N'| \leq C$.

Prueba 1.4.1 *Se crea un grafo $\hat{G} = (\hat{N}, \hat{E})$, donde $\hat{N} = \tilde{N} \cup \tilde{E}$ y $\tilde{E}$ se construye de la siguiente manera. Para cada par de nodos distintos $i, j \in \tilde{N}$, se crea un arista en $(i, j) \in \hat{E}$; para cada*

$e = (i, j) \in \tilde{E}$, se agrega (i, e) y (e, j) a $\hat{E}$. Para cada $i \in \tilde{N}$ su costo se establece como $c_i = 1$ y cero en otro caso. Para cada $e \in \tilde{E}$, se establece $f_e = 1$ y cero en otro caso. Se establece $N_i^{\alpha D} = \tilde{E}$ y $F_i = |\tilde{E}|$ para todo nodo $i \in \hat{N}$.

Se afirma que el VCP en $\tilde{G}$ tiene un límite superior de costo de C si y sólo si el EVCSP tiene una solución con un costo máximo C. Sea N' una cobertura de vértices de $\tilde{G}$ con $|N'| \leq C$ y sea H el subgrafo inducido de $\hat{G}$ con los nodos $N' \cup \tilde{E}$. Es fácil verificar que $|N_i^{\alpha D} \cap N(H)| = |\tilde{E}|$ y entonces $\sum_{j \in N_i^{\alpha D} \cap N(H)} f_i = |\tilde{E}| = F_i$. Cómo N' es una cobertura de vértices, cada $e = (i, j) \in \tilde{E}$ debe tener al menos uno de i y j en $\tilde{N}$ y entonces H debe contener una arista (e, k) para algún $k \in N'$. Además, $\tilde{N}$ forma un clique en $\hat{G}$. Por lo tanto, H debe ser conexo. Ya que cada $e \in \tilde{E} \subset \hat{N}$ no impone ningún costo, H tiene el mismo costo que N' en $\tilde{G}$. Por lo tanto, el EVCSP tiene una solución con un costo máximo C.

Se considera que un subgrafo inducido H es una solución del EVCSP. Se establece $N' = N(H) \cap \tilde{N}$. H contiene $\tilde{E}$: Como $f_j = 1$ para $j \in \tilde{E}$, para algún $i \in \hat{N}$, $F_i = |\tilde{E}|$ garantiza que $\tilde{E} \subset N(H)$.

Ya que H es conexo, cada $i \in N'$ debe tener una arista con un $e \in \tilde{E}$ en $\hat{G}$. Además, N' tiene a lo mucho C nodos. Por lo tanto, N' es una cobertura de vértices de $\tilde{G}$ con $|N'| \leq C$.

En la teoría de la complejidad computacional, un problema de decisión es NP-completo si éste está en la intersección de los conjuntos de problemas NP y NP-difícil. Con el Teorema 1.4.1, se tiene el siguiente corolario:

Corolario 1.4.1 *EVCSP es NP-difícil.*

Ya que el problema es NP-difícil, no existe una forma trivial de resolverlo, lo que hace al EVCSP computacionalmente interesante.

1.5 Hipótesis

Dado el planteamiento del problema y tomando en cuenta el desempeño de los algoritmos culturales en otros problemas relacionados (ver Sección 3.2), presentamos la siguiente hipótesis:

Es posible desarrollar un algoritmo cultural para resolver el problema de la ubicación de estaciones de carga de vehículos eléctricos, que tenga un desempeño competitivo en cuanto a la calidad de la solución y al tiempo computacional con respecto a los métodos actuales en el estado del arte.

1.6 Objetivos generales y particulares del proyecto

A continuación se mencionan los objetivos de este trabajo de investigación.

Objetivo general

Obtener un algoritmo cultural para resolver el problema de la ubicación de estaciones de carga de vehículos eléctricos minimizando el costo de construcción, que tenga un desempeño competitivo con respecto a los métodos actuales en el estado del arte.

Objetivos particulares

- Diseñar operadores y mecanismos *ad hoc* para mejorar el desempeño de un algoritmo evolutivo básico, basándonos en el estudio de las propiedades y características del problema.

- Analizar el desempeño del método propuesto así como de los diferentes componentes tomando en cuenta aspectos como la calidad de la solución, el tiempo computacional y el tamaño del problema.

- Obtener un estudio comparativo con al menos un algoritmo del estado del arte.

1.7 Organización de la tesis

El resto de esta tesis está organizada de la siguiente manera. En el Capítulo 2 se describe la terminología y los conceptos fundamentales relacionados al problema de investigación. Posteriormente, se presentan los trabajos relacionados con el tema de esta tesis, así como las aplicaciones de los algoritmos culturales en diferentes áreas en el Capítulo 3. En el Capítulo 4 se describen los componentes del algoritmo cultural propuesto. En el Capítulo 5 se describe la experimentación que se realizó, se presentan los resultados y el análisis de los mismos. Por último, se muestran las conclusiones de la tesis y el trabajo futuro en el Capítulo 6.

2

Marco teórico

A continuación se presentan algunos conceptos relevantes para el resto del documento. Principalmente se describen tanto la clasificación de los algoritmos evolutivos como los componentes del algoritmo cultural. Finalmente, se describen los conceptos acerca de teoría de grafos, los cuales fueron importantes para el diseño del algoritmo cultural propuesto.

2.1 Algoritmos evolutivos

Los algoritmos evolutivos son algoritmos estocásticos inspirados en los principios de la evolución natural.

Estos algoritmos operan sobre una población de individuos, donde cada individuo representa una solución potencial. Cada solución potencial se evalúa para obtener su aptitud. Posteriormente, mediante la selección de individuos se forma una nueva población, en la cual algunos miembros sufren variaciones mediante los operadores de variación para formar nuevas soluciones. Los principales operadores de variación son la mutación, la cual crea nuevos individuos mediante un pequeño cambio

en algunos de los individuos de la población, y el operador de cruza, quien crea nuevos individuos mediante la combinación de partes de dos individuos. Después de algún número de generaciones el algoritmo converge, y se espera que el mejor individuo represente una solución cercana al óptimo [7]. En el Algoritmo 1 se presenta la estructura del algoritmo evolutivo.

Algoritmo 1 Algoritmo evolutivo

1: $t \leftarrow 0$
2: Inicializar P { P es la población de individuos}
3: Evaluar P
4: **Mientras** $t < maxGen$ { $maxGen$ es el número máximo de generaciones} **Hacer**
5: $t + 1$
6: Seleccionar padres
7: Cruzar
8: Mutar
9: Evaluar
10: **Fin Mientras**
11: **Devolver** la mejor solución encontrada

Existen tres paradigmas principales de los distintos tipos de algoritmos evolutivos existentes [39]:

- Programación evolutiva.
- Estrategias evolutivas.
- Algoritmos genéticos.

2.1.1 Programación evolutiva

La programación evolutiva fue propuesta por Fogel [8]. La técnica es una abstracción de la evolución al nivel de las especies, por lo que no se requiere el uso de un operador de recombinación (diferentes especies no se pueden cruzar entre sí). Asimismo, usa selección probabilística. El algoritmo básico de la programación evolutiva es el siguiente:

- Generar aleatoriamente una población inicial.

- Se aplica mutación.
- Se calcula la aptitud de cada hijo y se usa un proceso de selección mediante torneo (normalmente estocástico) para determinar cuáles serán la soluciones que se retendrán.

Algoritmo 2 Algoritmo de programación evolutiva

1: $t \leftarrow 0$
2: Inicializar P { P es la población de individuos}
3: Evaluar P
4: **Mientras** $t < maxGen$ { $maxGen$ es el número máximo de generaciones} **Hacer**
5: $t \leftarrow t + 1$
6: Seleccionar padres
7: Mutar
8: Evaluar
9: **Fin Mientras**
10: **Devolver** la mejor solución encontrada

2.1.2 Estrategias evolutivas

Schwefel [41, 42] introdujo las estrategias evolutivas denominadas $(\mu + \lambda)$ y (μ, λ). La notación se refiere al mecanismo de selección utilizado: en el primer caso, sobreviven los μ mejores individuos obtenidos de la unión de padres e hijos; en el segundo caso, sólo sobreviven los μ mejores hijos a la siguiente generación.

El operador principal es el de mutación, donde a cada elemento del vector se le aplica una variación obtenida de forma aleatoria siguiendo una distribución normal. El operador secundario es el de cruza.

2.1.3 Algoritmos genéticos

Los algoritmos genéticos fueron desarrollados por Holland en 1962 [14]. El operador principal del algoritmo genético es la cruza y el secundario es la mutación, usa selección probabilística. En el Algoritmo 3 se observa la estructura del algoritmo genético básico.

Algoritmo 3 Algoritmo genético

1: $t \leftarrow 0$
2: Inicializar P { P es la población de individuos}
3: Evaluar P
4: **Mientras** $t < maxGen$ { $maxGen$ es el número máximo de generaciones} **Hacer**
5: $t \leftarrow t + 1$
6: Seleccionar padres
7: Cruzar
8: Mutar
9: Evaluar
10: Elitismo
11: **Fin Mientras**
12: **Devolver** la mejor solución encontrada

2.2 Manejo de restricciones

Muchos de los problemas de optimización tienen restricciones y no es trivial su manejo. Las funciones de penalización se usan frecuentemente en la optimización con restricciones, dichas funciones son controladas por un coeficiente de penalización, encontrarlo adaptativamente es equivalente a ordenar individuos en una población adaptativamente, sin embargo, es complicado ordenar a los individuos de acuerdo con sus valores de penalización y de la función objetivo.

En general, existen tres diferentes categorías en las que se pueden clasificar los métodos de penalización. En la primera categoría, el valor del coeficiente de penalización es muy pequeño para influenciar el ordenamiento de los individuos, por lo que todas las comparaciones se basan solamente en la función de aptitud. Esta categoría es llamada *sub-penalización*. En la segunda categoría, el valor del coeficiente de penalización es tan grande que se puede ignorar el efecto de la función objetivo, por lo que todas las comparaciones se basan solamente en la función de penalización. Esta categoría es llamada *sobre-penalización*. En la tercera categoría, todas las comparaciones se basan en una combinación de la función objetivo y la función de penalización [38].

La técnica tradicional de manejo de restricciones utilizada en las estrategias de evolución se sitúa aproximadamente en la categoría de *sobrepenalización*, ya que todos los individuos inviables

se consideran peores que los viables [5, 21, 43]. Los tipos de restricciones pueden ser: lineales o no lineales y de igualdad o desigualdad.

Las estrategias para el manejo de restricciones se pueden clasificar como: estrategias de rechazo, estrategias de penalización, estrategias de reparación y estrategias de decodificación [48].

2.2.1 Estrategias de rechazo

En las estrategias de rechazo las soluciones factibles se conservan durante la búsqueda y luego las soluciones infactibles son descartadas automáticamente. Este tipo de estrategias son concebibles si la porción de soluciones infactibles del espacio de búsqueda es muy pequeño.

2.2.2 Estrategias de penalización

En las estrategias de penalización la función objetivo sin restricciones se extiende mediante una función de penalización que castigará a las soluciones infactibles.

A continuación, se describe la estrategia de penalización que se utilizó para el manejo de las restricciones del problema del EVCSP.

2.2.2.1. Jerarquías estocásticas

Es un enfoque que balancea la contribución entre la violación de restricciones y la función objetivo de forma estocástica. Tal balance se implementa comparando en el P_f por ciento de los casos únicamente con la violación de restricciones, mientras que el resto de las veces la comparación se hace sólo con la función objetivo, donde P_f es un parámetro de entrada. Usa un procedimiento de tipo burbuja, donde λ individuos se ordenan comparando a individuos adyacentes en al menos λ pasadas [38]. En el Algoritmo 4 se muestra el pseudocódigo del enfoque de jerarquías estocásticas, donde x^j es el j-ésimo individuo de la población, ϕ es una función de valor real que impone una penalización en caso de que el individuo sea infactible (en otro caso es cero), f es la función objetivo,

$U(0,1)$ es una función que devuelve un número aleatorio uniforme y nb es el número de barridos que atraviesan toda la población.

Algoritmo 4 Jerarquías estocásticas

```
1: i = 1
2: Repetir
3:    Para j = 1 to λ − 1 Hacer
4:       muestrear u ∈ U(0, 1)
5:       Si (φ(x^j) = φ(x^{j+1}) = 0) or (u < P_f) Entonces
6:          Si (f(x^j) > f(x^{j+1})) Entonces
7:             intercambiar(x^j, x^{j+1})
8:          Fin Si
9:       Si no
10:         Si (φ(x^j) > φ(x^{j+1})) Entonces
11:            intercambiar(x^j, x^{j+1})
12:         Fin Si
13:      Fin Si
14:   Fin Para
15:   Si no hay intercambios Entonces
16:      break
17:   Fin Si
18:   i = i + 1
19: Hasta que i ≥ nb
```

2.2.3 Estrategias de reparación

Consisten en algoritmos heurísticos que transforman una solución infactible en una factible. Estas estrategias se aplican en el caso donde los operadores de búsqueda utilizados por los algoritmos de optimización pueden generar soluciones infactibles.

2.2.4 Estrategias de decodificación

El procedimiento puede ser visto como una función $R \to T$, donde T es el conjunto de soluciones factibles y R es el conjunto de todas las soluciones, que asocia con cada representación $r \in R$ una solución factible $t \in T$ en el espacio de búsqueda. Esta estrategia consiste en usar codificaciones indirectas. La topología del espacio de búsqueda se transforma usando una función de decodificación.

2.3 Algoritmo cultural

Los algoritmos culturales se basan en algunas teorías originadas en sociología y arqueología, las cuales intentan modelar la evolución cultural. El proceso de evolución cultural se ha modelado bajo las perspectivas de micro-evolución y macro-evolución. La primera consiste en la transmisión genética de comportamientos y rasgos entre los individuos de una población, mientras que la segunda se describe en términos de la formación de creencias generalizadas basadas en la experiencia individual; dichas creencias sirven para guiar el comportamiento de los individuos que pertenecen a una población. Los sistemas culturales soportan la transmisión de información tanto a nivel individual como a nivel grupal.

Los algoritmos culturales operan en dos espacios. El primero es el espacio de la población, que es el conjunto de individuos. Cada individuo tiene un conjunto de características independientes que se usan para determinar su aptitud. A través del tiempo, tales individuos se pueden reemplazar por algunos de sus descendientes, los cuales se obtienen a través de la aplicación de un conjunto de operadores a la población. El segundo espacio es el de las creencias, donde es almacenado el conocimiento adquirido por los individuos a través de generaciones. A su vez, dicho conocimiento se puede usar para modificar el comportamiento de los individuos [25, 34].

Un algoritmo cultural se define como una óctupla [35] y sus componentes se listan a continuación:

$$CA = \langle P, S, V_c, f, B, Aceptación, Actualización, Influencia \rangle$$

P: es la población.

S: es el operador de selección.

V_c: son los operadores de variación (tal como cruza o mutación).

f: es la función de aptitud.

B: es el espacio de creencias.

$Aceptación$: es la función de aceptación que determina qué individuos de la población actual son capaces de impactar, o ser candidatos para contribuir, a las creencias actuales.

$Actualización$: es un operador del espacio de creencias que ajusta o actualiza el conocimiento de dicho espacio de creencias, B.

$Influencia$: es un conjunto de funciones para sesgar el operador de variación V_c de acuerdo con el contenido del espacio de creencias.

$Aceptación$ e $Influencia$ juntas representan el protocolo de comunicación para un algoritmo cultural. En la Figura 2.1 se muestran los espacios de un algoritmo cultural y en el Algoritmo 4 se presenta la estructura de dicho algoritmo.

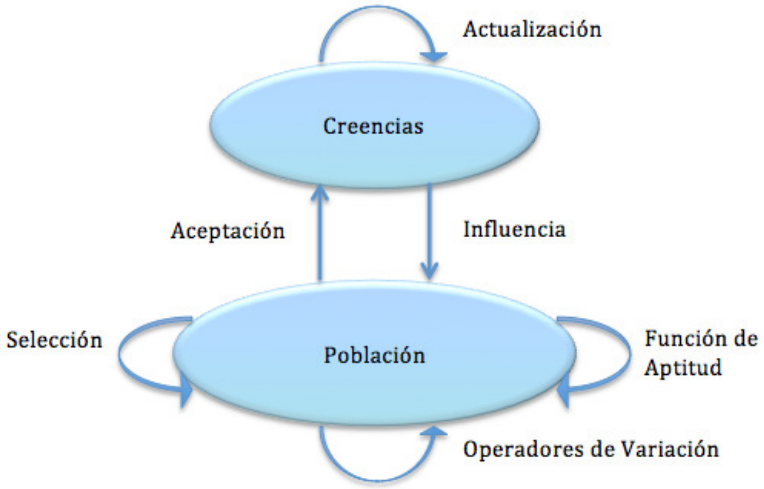


Figura 2.1: Espacios de un algoritmo cultural

Algoritmo 5 Algoritmo cultural

1: $t \leftarrow 0$
2: Inicializar P { P es la población de individuos}
3: Evaluar P
4: Inicializar BS { BS es el espacio de creencias}
5: **Mientras** $t < maxGen$ { $maxGen$ es el número máximo de generaciones} **Hacer**
6: $t + 1$
7: Seleccionar padres
8: Actualizar BS (con los individuos aceptados de P)
9: Aplicar los operadores de variación (bajo la influencia de BS)
10: Evaluar
11: Elitismo
12: **Fin Mientras**
13: **Devolver** la mejor solución encontrada

2.3.1 Fuentes de conocimiento

Algunas de las principales fuentes de conocimiento empleadas en los algoritmos culturales son [24]:

- **Conocimiento de dominio**: consiste en utilizar el conocimiento específico del problema para sesgar el proceso de resolución del mismo.
- **Conocimiento situacional**: consiste del mejor ejemplar encontrado a lo largo del proceso evolutivo. Se emplea como líder, de modo que los individuos seleccionados para aplicarles los operadores de variación terminen siendo más similares a él. El conocimiento situacional se actualiza si el mejor individuo de la generación actual ha cambiado.
- **Conocimiento normativo**: contiene los intervalos para cada parámetro que determinan el rango de lo que es considerado como una buena solución, con el fin de mover nuevas soluciones hacia estos intervalos.
- **Conocimiento topográfico**: consiste en crear un mapa del paisaje de búsqueda del problema durante el proceso evolutivo. Éste consiste de un conjunto de celdas y el mejor individuo encontrado en cada celda. El conocimiento topográfico también tiene una lista ordenada de las mejores celdas, basándose en el valor de aptitud del mejor individuo en cada una de ellas.
- **Conocimiento histórico**: el conocimiento histórico registra, en una lista, la ubicación del mejor individuo encontrado antes de cada cambio de entorno.

2.4 Algoritmos voraces

Los algoritmos voraces reciben su nombre del hecho de que estos siempre hacen una elección que parece la mejor solucion posible en este momento, tomando decisiones basadas en un criterio único, en lugar de un análisis global, y sin tomar en cuenta las consecuencias que pueden resultar de esa

elección. Los algoritmos voraces son a menudo intuitivamente simples y fácil de implementar. Por lo tanto, los algoritmos voraces son muy populares para resolver problemas de optimización [50].

En este trabajo de tesis se comparan los resultados de un algoritmo genético simple y del algoritmo cultural propuesto con los de un algoritmo voraz, el cual se implementó de acuerdo con el trabajo de Lam et al. [22].

2.5 Conceptos relevantes de grafos

A continuación se listan los conceptos que fueron importantes para el diseño de los operadores del algoritmo propuesto.

Grafo conexo: Sea $G = \langle V, E \rangle$ un grafo no dirigido. Se dice *conexo* si existe camino entre todo par de nodos [11].

Componentes conexos: Se llama componente conexo de G a todo subgrafo maximal conexo de G [31].

Puntos de articulación: Decimos que un nodo es un punto de articulación fuerte si su remoción aumenta el número de componentes fuertemente conexos de G [18].

2.6 Resumen del capítulo

En este capítulo se mencionaron varios conceptos relacionados al diseño del algoritmo cultural propuesto. En primera instancia, se describieron los tres paradigmas principales de los diferentes tipos de algoritmos evolutivos existentes, incluyendo al algoritmo genético. Posteriormente, se describieron las estrategias para el manejo de restricciones de los problemas de optimización. Por último, se explicaron cada uno de los componentes y la estructura del algoritmo cultural; y se mencionaron conceptos relevantes de grafos.

3

Estado del arte

En este capítulo se presenta una breve descripción de los trabajos relacionados al EVCSP; así también, se muestran diferentes métodos con los que ha sido abordado. Finalmente, algunos ejemplos de las áreas de aplicación de los algoritmos culturales.

3.1 Trabajos relacionados

El problema del EVCSP se ha abordado recientemente empleando diversos métodos de optimización. A continuación se describen brevemente algunos de los métodos utilizados para afrontar el problema considerando aquellos en los que se involucran funciones de costo con beneficio económico.

Uno de los trabajos en el que el objetivo es encontrar las ubicaciones óptimas reduciendo diferentes tipos de costos fue propuesto por Liu et al. [27], donde se construyó un problema de optimización a través del cual se minimizaron los costos de construcción, de operación y de carga con las restricciones de flujo de tráfico y de requerimientos de carga. Así también se adoptó el método de optimización

basado en cúmulo de partículas (PSO por sus siglas en inglés) para calcular la solución del problema. Para cada estación designaron su respectiva capacidad de servicio, costo de construcción, cantidad de cargadores y espacio de suelo. Como caso de estudio establecieron 4 tipos de niveles de estaciones de carga tomando en cuenta las especificaciones técnicas de estaciones de carga en Pekín. Cuando las estaciones de carga se ubican en el nivel 4 se requieren al menos 14 estaciones de carga y cuando se ubican en el nivel 1 se necesitan al menos 3. Definen una variable llamada variable de circulación, que es en esencia el número de estaciones de carga n en un rango de 3 a 14. Para cada valor de n calcularon el costo de inversión y de operación. Compararon los resultados obtenidos de un algoritmo PSO básico con los del algoritmo propuesto PSO adaptado, con respecto al valor promedio de todos los resultados de las 50 ejecuciones independientes. Los resultados mostraron que regular automáticamente el tamaño del factor de inercia w, puede acelerar la tasa de convergencia y mejorar el desempeño del algoritmo PSO básico.

En el trabajo de Jia et al. [20] se estudiaron los problemas de ubicación y de tamaño, donde las ubicaciones y el número de cargadores en cada estación se determinaron al mismo tiempo tomando en cuenta la demanda de carga. Se resolvió el problema con datos de Estocolmo, Suecia, empleando una herramienta para resolver problemas de optimización llamada CPLEX [4]. Simplificaron la estructura de la red vial de Estocolmo colocando un nodo en cada uno de sus 12 distritos, y de acuerdo a las conexiones entre ellos establecieron 13 caminos. A partir de los resultados se observó que las estaciones de carga están principalmente concentradas en aquellos nodos con gran demanda, debido a que cuanto más cerca están las estaciones de carga de los nodos, menor es el tiempo y los costos de energía. Entonces los usuarios de EV necesitan pasar por una distancia más larga hacia la estación para realizar la carga, lo que resulta en mayores costos generales. En este trabajo no toman en cuenta la distribución de las estaciones de carga para la conveniencia de los conductores y la accesibilidad de los EV. Falta tomar en cuenta el tiempo computacional de la solución con respecto al tamaño del problema.

Para la solución del problema de la ubicación de las estaciones de carga del EV también se han empleado adaptaciones a algoritmos genéticos como en el trabajo presentado por Mehar and Senouci [30] donde proponen una formulación matemática. Minimizan los costos de inversión y de transporte para encontrar las ubicaciones óptimas. Tienen una fase de preproceso para crear grupos a partir de la información del tráfico y calculan las demandas de carga, lo cual les ayuda a reducir la complejidad del problema y los tiempos de ejecución. El caso de estudio se generó con un área de 64 kilómetros cuadrados. Generaron diferentes escenarios teniendo como tamaño inicial de la población 100 cromosomas. Comparan su propuesta con una solución exacta generada con CPLEX tomando en cuenta el tiempo de ejecución. Los resultados confirmaron que al introducir un "operador de permutación" al algoritmo genético ayuda a respetar la restricción del problema y a converger a la solución óptima. Falta el estudio de la complejidad de su contribución.

Otros métodos como los algoritmos genéticos híbridos se han empleado para encontrar las mejores ubicaciones, como en el trabajo de You and Hsieh [55] en el que minimizan los costos de inversión y de viaje. Los casos de estudio oscilan entre 15 y 80 nodos. En términos de la calidad de la solución y eficiencia computacional mejoran los resultados comparándose con la solución que obtuvieron empleando CPLEX.

Xiong et al. [52] se enfocan en el comportamiento estratégico del conductor del EV para minimizar el costo de sus cargas, además toman en cuenta factores como las condiciones de tráfico de la red vial y los tiempos de espera en las estaciones de carga. Como caso de estudio dividen el área de 718 kilómetros cuadrados en 23 zonas. Proponen un nuevo algoritmo heurístico para resolver el problema y emplean los datos de Singapur para realizar experimentos. Los resultados de la experimentación muestran que su enfoque obtiene ubicaciones efectivas de las estaciones de carga y supera los métodos tradicionalmente usados como base comparativa, es decir, asignar estaciones de carga considerando la demanda de carga en cada zona y basándose en las condiciones de tráfico.

Islam et al. [16] emplean un nuevo algoritmo de optimización heurística para determinar la ubicación óptima y tamaño de las estaciones de carga en los sistemas de red de carreteras de

Malasia. La formulación del problema de optimización se basa principalmente en los costos de operación, de tierra, de transporte y de la pérdida de energía de la subestación. La técnica empleada de optimización fue comparada con un algoritmo genético y un PSO para evaluar su efectividad. Para validar los resultados, todas las técnicas se ejecutaron 50 veces. Los resultados mostraron que el método propuesto requiere menos ejecuciones que el algoritmo genético y el PSO para encontrar la solución óptima, además, fue más consistente que el algoritmo genético.

Lam et al. [22] formulan el problema de EVCSP en términos del costo de construcción y utilizando como restricciones la demanda y la autonomía de los autos. Además proponen cuatro métodos de solución. Examinaron la calidad de la solución, el aumento del tiempo computacional con respecto al crecimiento del tamaño del problema y probaron el desempeño de los métodos de solución con varios casos sintéticos y un escenario real (Hong Kong, China). Los casos de prueba oscilan entre 10 y 200 nodos. Este trabajo es relevante debido a que se incluye un análisis de complejidad, en el cual se mostró que el problema es NP-completo. Esto le da una nueva dimensión al problema ya que, además de las implicaciones prácticas de su solución, se muestra como un problema computacionalmente interesante desde el punto de vista abstracto.

A continuación se describen los métodos empleados:

- Método I: Programación lineal entera-mixta iterativa [45] (I MILP por sus siglas en inglés). Para resolver el problema requieren aplicar iterativamente MILP a todos los nodos. El tiempo computacional para resolver MILP crece exponencialmente con el tamaño del problema n. Emplean las herramientas CPLEX y YALMIP [28] para calcular la solución, los cuales producen una solución óptima.

- Método II: Enfoque voraz. Asumen que cuentan con una solución factible del problema. Construyen un algoritmo voraz reduciendo lo más posible el costo total en cada iteración obteniendo una solución sub-óptima.

- Método III: Programación lineal entera-mixta efectiva (E MILP por sus siglas en inglés). Con

Referencia	Función de costo	Restricciones	Caso de estudio	Análisis de complejidad	Métodos	Ventajas	Desventajas
Liu et al. (2012) [27]	Construcción, operación y carga	Flujo de tráfico y demanda de carga	Real, 3-14 nodos, topología de red	×	PSO	Solución cercana al óptimo	Convergencia prematura
Jia et al. (2012) [20]	Construcción y operación	Demanda de carga y Estructura de la red vial	Real, 12 nodos y 13 aristas, topología de red	×	ILP	Resuelve diversas combinaciones de problemas	Sólo trabaja con funciones lineales
Mehar and Senouci (2013) [30]	Inversión y transporte	Demanda de carga	Real, 16 nodos, topología de red	×	GA	El método más empleado para problemas de ubicación	Toma mucho tiempo
Lam et al. (2014) [22]	Construcción	Demanda de carga	Sintético y real, 10-200 nodos,18 nodos, topología de red	✓	I MILP, E MILP, voraz y CRO	Estudio comparativo entre los métodos empleados	CRO sólo muestra un mejor tiempo computacional al emplear al método voraz como solución inicial

Tabla 3.1: Comparación de los métodos empleados en el estado del arte.

este método se pudo minimizar el tiempo computacional con la elección del nodo de menor grado en el subgrafo. De este modo se pudo simplificar el Método I mediante la explotación de la estructura de la red del grafo. Las soluciones de los métodos I y III son equivalentes.

- Método IV: Optimización por reacción química (CRO por sus siglas en inglés). Emplean el Método II y una modificación del mismo como solución inicial del algoritmo. En términos de calidad de solución el Método IV es mejor que el Método II, sin embargo éstos no pueden producir soluciones óptimas, sobre todo cuando el espacio de soluciones es cada vez mayor.

Los Métodos I y III siempre dan las mejores soluciones, es decir, son óptimos. En términos del tiempo computacional, el Método II es el más rápido, el Método III ocupa el segundo lugar, el Método IV el tercero y por último el Método I.

En la Tabla 3.1 se muestra una comparación entre los diferentes métodos de optimización que abordan el problema de EVCSP.

Los algoritmos culturales no se han empleado para resolver este tipo de problema y debido a su éxito en otros problemas similares, en este trabajo de tesis proponemos un algoritmo cultural ad hoc, con el que buscamos explotar sus ventajas mostradas en otros trabajos. Se utilizará la versión del

problema propuesta en [22] debido a que realizan un análisis de la complejidad del problema de EVCSP. Por la misma razón, los resultados experimentales se comparan con los mostrados en tal trabajo.

3.2 Aplicaciones de los algoritmos culturales

Los algoritmos culturales son un sistema de herencia dual donde la evolución toma lugar tanto a nivel de la población como a nivel de adquisición de creencias. Los dos componentes interactúan a través de un protocolo de comunicación, el cual determina el conjunto de individuos aceptables que son capaces de actualizar el espacio de creencias. Igualmente el protocolo determina cómo las creencias actualizadas pueden impactar la adaptación del componente de la población.

El espacio de creencias en los algoritmos culturales es un repositorio para la información referente al proceso de auto-adaptación. La información almacenada en el espacio de creencias puede relacionarse con varios parámetros globales que se pueden utilizar para dirigir la población a través del protocolo de comunicación especificado. En la siguiente sección se describen, en el orden de desarrollo, las áreas más relevantes a las que se han aplicado algoritmos culturales [35].

3.2.1 Desarrollo histórico de los algoritmos culturales

En cuanto a la programación de recursos y la evolución de la agricultura, se empleó un modelo inspirado en el componente poblacional usado por Reynolds [1] y el algoritmo genético de Goldberg and Holland [10] para modelar la evolución de la agricultura en el Valle de Oaxaca, México. El modelo muestra que la agricultura podría haber evolucionado con la acumulación de decisiones de planificación de recursos a pequeña escala a través de largos periodos de tiempo en el Valle.

Hablando de las aplicaciones de aprendizaje de conceptos, una segunda generación de modelos se usó para adquirir conocimiento sobre la estructura de los elementos de la población en adición a la información referente a los parámetros de la población [37]. El objetivo es aprender conceptos generales sobre aspectos de los individuos en una población.

En la optimización de funciones con valores reales, la tercera generación de modelos se desarrolló para trabajar con algoritmos genéticos con valores reales tales como los propuestos por Reynolds et al. [36]. Aquí, las características asociadas con los individuos en la población fueron valores reales. Cada una de las características en la población corresponde a un intervalo de valores reales en el espacio de creencias.

3.2.2 Aplicación de los algoritmos culturales en problemas combinatorios

Wang et al. [51] diseñan una estrategia de control de vibración óptima para la suspensión activa de un vehículo basándose en un algoritmo cultural. Los resultados mostraron que la estrategia de control empleando un algoritmo cultural reduce significativamente la vibración de la carrocería.

Otra área de aplicación se describe en el trabajo de Yan et al. [54], donde emplean un algoritmo cultural para resolver el problema de optimización del diseño de circuitos digitales combinacionales. Los resultados del experimento mostraron que el algoritmo es eficaz para este problema.

Para el caso de realizar la simulación transitoria de las redes de gas natural con presiones de entrada y salida conocidas, se empleó un algoritmo cultural en Madoliat et al. [29], donde el enfoque

propuesto reduce la complejidad de resolver las ecuaciones de flujo transitorio de la red, mientras que los resultados confirman su precisión y eficiencia.

En la siguiente subsección, se describen algunos ejemplos de las aplicaciones de los algoritmos culturales a problemas similares al EVCSP.

3.2.3 Aplicación de los algoritmos culturales en problemas de asignación de recursos

Haldar and Chakraborty [12] emplearon un algoritmo cultural modificado para la reducción de pérdida de potencia real mediante la asignación óptima de condensadores. El algoritmo evolutivo proporcionó resultados prometedores afrontando la reducción de la perdida de energía real en sistemas de distribución. Se observó que los conocimientos situacional e histórico influyen en el espacio de la población con la actualización del espacio de creencias.

El emparejamiento de imágenes es un problema diferente en el que el algoritmo cultural también ha demostrado tener un gran efecto en lo referente a la convergencia, precisión y robustez [53].

Soza et al. [46] usan un algoritmo cultural para el problema de horarios. Los resultados obtenidos con el enfoque propuesto mejoraron a los resultados de dos algoritmos evolutivos y un recocido simulado con los que se compraron.

Para el problema de *job shop scheduling* se usó un algoritmo cultural que fue capaz de producir resultados competitivos con respecto a los siguientes algoritmos: procedimiento de búsqueda adaptable aleatorizado y (GRASP por sus siglas en inglés), un GRASP paralelo, un algoritmo genético, un algoritmo genético híbrido y un método determinista llamado *shifting bottleneck* a un costo computacional significativamente menor que al menos uno de ellos y sin utilizar ningún tipo de procesamiento paralelo [3].

3.3 Resumen del capítulo

En este capítulo se describieron algunos de los trabajos encontrados en la literatura que han intentado resolver el problema del EVCSP. Cada uno de ellos aborda el problema con diferentes funciones de costo y restricciones. También se presentó un resumen de las principales técnicas con las que el EVCSP ha sido abordado. Por último, se mencionaron algunos casos de éxito de los algoritmos culturales en diferentes áreas de aplicación, lo cual indica que este algoritmo evolutivo obtiene soluciones de calidad para diferentes problemas combinatorios.

4

Método propuesto

En este capítulo se describen las metas que se cumplieron para el desarrollo de la tesis, así como también el manejo de las restricciones del problema y los elementos del algoritmo genético simple. Finalmente, se detalla el diseño del algoritmo cultural propuesto, así como cada uno de sus componentes.

4.1 Descripción de la metodología

A continuación, se describen los pasos que se siguieron para cumplir con los objetivos de esta tesis.

La primera meta que se alcanzó fue el estudio y análisis del estado del arte, el cual se enfocó en los métodos de optimización empleados para solucionar problemas similares al EVCSP, considerando aquellos procedimientos con los que se obtienen ahorros en costos de construcción de estaciones de carga. Sin embargo, el estudio del estado del arte se realizó a lo largo de todo el trabajo de tesis.

Una vez finalizada la primera meta, se definió el manejo de las restricciones del problema y se realizó experimentación con un algoritmo genético simple.

Para realizar el diseño del algoritmo cultural se investigaron las principales características del problema, se definieron las fuentes de conocimiento, la función de aceptación y la función de influencia que emplea el algoritmo cultural, con el propósito de definir los aspectos principales de la propuesta de este trabajo.

Posteriormente, se realizó la experimentación con diferentes tamaños del problema y con diferentes configuraciones en el algoritmo. Con los resultados obtenidos se evaluó el desempeño del algoritmo cultural y se validó si es una alternativa viable para resolver el problema tomando en cuenta aspectos como la calidad de la solución, el tiempo computacional y el tamaño del problema.

Por último, se compararon los resultados obtenidos con el algoritmo genético simple y el algoritmo voraz del estado del arte. Así también, se realizó un estudio de significación estadística para comparar los resultados entre el algoritmo genético y el cultural propuesto.

4.2 Manejo de restricciones del problema

Dado que el algoritmo genético opera como una técnica de optimización sin restricciones, es necesario diseñar algún mecanismo que permita incorporar información sobre la violación de restricciones en la función de aptitud.

Con el objetivo de conocer el grado de violación de cada restricción del problema, se definieron las siguientes funciones de penalización:

Para el caso de la restricción de conectividad del subgrafo inducido H del grafo $\hat{G}$ se define la Función 4.1.

$$Y(x) = \left(\sum_{i=0}^{n} x_i\right) - v \tag{4.1}$$

donde v es el número de nodos visitados. Para obtener el valor de v, se implementó un algoritmo de búsqueda en anchura (BFS, por sus siglas en inglés), debido al modo en que realiza la exploración en un grafo, el cual se asegura de visitar los nodos en orden creciente de su distancia desde algún nodo inicial, además, tiene la propiedad de que todas sus rutas desde algún nodo inicial son lo más cortas posibles. Por estas razones, se consideró que era el adecuado para este tipo de problema en el que se manejan distancias más cortas entre nodos.

Con BFS, dado un nodo inicial s (seleccionado aleatoriamente, de entre aquellos nodos i con $x_i = 1$), se visitaron los nodos del subgrafo H para descubrir todos los nodos alcanzables desde s. Por consiguiente, para garantizar que se visitaron todos los nodos i con $x_i = 1$ desde el nodo s en H, se obtiene la diferencia entre el total de nodos con $x_i = 1$ y el total de nodos visitados v, si la diferencia es cero, implica que la restricción de conectividad se satisface. En otro caso, indica la magnitud de la violación de la restricción de conectividad $Y(x)$.

Para calcular la magnitud de la violación de la restricción de demanda de las estaciones de carga, se plantea la Función 4.2:

$$W(x) = \sum_{i=0}^{n} \left(F_i - \left(\sum_{j \in N_i^{\alpha D}} f_j x_j \right) \right) \tag{4.2}$$

donde $\sum_{j \in N_i^{\alpha D}} f_j x_j$ es la capacidad total, definida a partir de los nodos $j \in N'$; y F_i es la demanda del nodo i, las cuales ya se definieron en las Secciones 1.3.1 y 1.3.2. Para cada nodo $i \in N$, cada vez que se viola la restricción de demanda, es decir, siempre que $\sum_{j \in N_i^{\alpha D}} f_j x_j < Fi$, se calcula la suma acumulada de las diferencias entre las mismas.

Dadas la función objetivo $f(x)$ en la Ecuación 4.3, la cual se definió en la Sección 1.3.2, y las magnitudes de violación de las funciones $Y(x)$ y $W(x)$, se empleó el enfoque de jerarquías estocásticas, como en el Algoritmo 4, para balancear la contribución entre la violación de restricciones $\phi(x)$ en la Ecuación 4.4 y la función objetivo $f(x)$ de forma estocástica.

$$f(x) = \sum_{i=1}^{n} c_i x_i \tag{4.3}$$

$$\phi(x) = Y(x) + W(x) \tag{4.4}$$

La probabilidad que se empleó para usar solamente la función objetivo, en vez de usar la violación de restricciones, durante las comparaciones en el ordenamiento, (en caso de que ambas soluciones a comparar sean infactibles) fue $P_f = 0.23$, debido a que se tiene interés en las soluciones factibles como soluciones finales, P_f debe ser menor a 0.5 de modo que haya un sesgo en contra de soluciones infactibles.

Una vez ordenada la población, se le asignó un valor de aptitud $fit(x)$ a cada individuo de acuerdo con su posición en la lista ordenada, asignando el valor de 30 (la cantidad de individuos de la población), como la aptitud más alta, al individuo en la primera posición.

4.3 Algoritmo genético simple

Se utilizó un algoritmo genético simple como el Algoritmo 3, con el propósito de cubrir el componente poblacional para el diseño del algoritmo cultural propuesto, así como también, para observar el comportamiento del problema del EVCSP bajo el enfoque del cómputo evolutivo. El algoritmo genético simple utiliza tres tipos de operadores: selección, cruza y mutación. Se utilizó reemplazo generacional (donde los padres son descartados y reemplazados por los hijos), el cual es una estrategia comúnmente usada para el diseño de algoritmos genéticos simples [7] y elitismo como un mecanismo necesario para la convergencia.

En las siguientes secciones se mencionan tanto el tipo de representación como los operadores evolutivos utilizados en el algoritmo genético.

4.3.1 Representación y operadores evolutivos

Dado que la función objetivo en la Ecuación 4.3 emplea la variable de decisión booleana x_i para indicar que el nodo i se elige para construir una estación de carga, se utilizó la representación

binaria para los cromosomas de los individuos de la población del algoritmo genético. Un ejemplo de la representación binaria para un individuo con 10 genes se muestra en la Figura 4.1, donde las posiciones con valores igual a uno indican los nodos seleccionados para construir estaciones de carga.

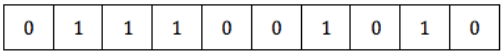


Figura 4.1: Ejemplo de un individuo con 10 genes

Los operadores evolutivos son indispensables para mantener la diversidad genética en la población y para determinar el modo en que el algoritmo explora el espacio de búsqueda. A continuación se describen los operadores evolutivos que se emplearon en el algoritmo genético.

4.3.1.1. Operador de selección de padres

Con el operador de selección es importante obtener un balance entre la posibilidad de que se elijan los mejores individuos y los menos aptos, en el sentido de que los individuos menos aptos pueden incluir material genético útil para el proceso de reproducción. Esto con el fin de mejorar las soluciones a través de las generaciones al mismo tiempo que se mantiene la diversidad .

En el algoritmo genético que se empleó en este trabajo de tesis se utilizó un operador de selección por ruleta [19]. La característica relevante de este método de selección es el hecho de que da a cada individuo x^j de la población actual una probabilidad $p(x^j)$ de ser seleccionado, proporcional a su aptitud $fit(x^j)$. Esto se observa en la Ecuación 4.5.

$$p(x^j) = \frac{fit(x^j)}{\sum_{j=1}^{|P|} fit(x^j)} \tag{4.5}$$

donde $|P|$ denota el tamaño de la población en términos del número de individuos.

4.3.1.2. Operador de cruza

El propósito de la cruza es crear nuevos descendientes mediante el intercambio de información entre individuos de la población. Esto se realiza mediante la selección aleatoria de dos individuos de la población y algunas porciones se intercambian entre los cromosomas [33]. En el algoritmo genético simple de este trabajo de tesis, se utilizó la cruza de dos puntos, en la cual se eligen aleatoriamente dos puntos de cruza a lo largo de la longitud de los cromosomas para intercambiar las partes alternando entre los cromosomas de los dos individuos. En la Figura 4.2 se muestra la cruza de dos puntos.

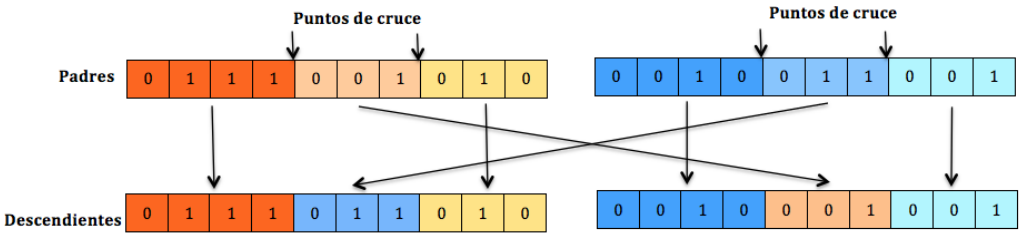


Figura 4.2: Ejemplo de cruza de 2 puntos

4.3.1.3. Operador de mutación

El propósito de la mutación es generar un individuo en el vecindario del individuo actual. Esto se realiza para evitar una pérdida prematura de material genético importante en una posición particular, y para mantener la diversidad de la población [33]. El operador de mutación cambia algunos valores en el cromosoma del individuo. Se pueden utilizar varios métodos para implementar al operador de mutación. En esta tesis se utilizó la mutación uniforme, en la cual se selecciona un número aleatorio uniforme entre 0 y 1 para cada dígito binario en el cromosoma del individuo, si el número aleatorio es menor a cierta probabilidad de mutación, entonces el dígito binario del cromosoma cambia. De otro modo, el dígito binario permanece intacto. En la Figura 4.3 se muestra un ejemplo del funcionamiento de la mutación uniforme.

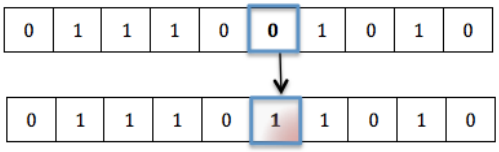


Figura 4.3: Ejemplo de mutación uniforme

4.4 Componentes del algoritmo cultural

Los componentes de un algoritmo cultural se muestran en la Figura 2.1, donde se observan tanto el espacio poblacional, el cual se cubrió con un algoritmo genético simple, como el espacio de creencias. Estas creencias se pueden usar como guías para dirigir la evolución de la población. Las fuentes de conocimiento que se encuentran en el espacio de creencias sirven de ayuda para mejorar el proceso evolutivo del algoritmo. El algoritmo cultural propuesto toma como base el algoritmo genético simple, utilizando los mismos siguientes componentes: selección por ruleta, cruza de dos puntos, mutación uniforme, reemplazo generacional y elitismo.

4.4.1 Diseño de los operadores utilizados

Después de entender el problema del EVCSP y de conocer las cinco principales fuentes de conocimiento de un algoritmo cultural, la primera pregunta planteada fue: ¿cuáles de las fuentes de conocimiento se deberían utilizar para realizar el proceso de resolución del problema?

En principio, se identificó el conocimiento sobre las características del problema y las restricciones del mismo. Se tomó la decisión de que las primeras fuentes de conocimiento a utilizar fueran la de dominio y la situacional, ya que se contaba con el conocimiento requerido por las mismas; es decir, por un lado, la importancia de mantener la conectividad del subgrafo H y de satisfacer la demanda de las estaciones de carga, y por otro lado, el conocimiento a priori para la aplicación del espacio de creencias situacional.

La siguiente pregunta fue: ¿cuál sería una manera apropiada de influir en los individuos utilizando

el conocimiento del dominio del problema para obtener soluciones adecuadas? En el contexto del EVCSP al ser este un problema de minimización de costos, la idea de considerar quitar de la solución a aquellos nodos con los costos más grandes parece plausible; sin embargo, para que esto tenga sentido se tendría que verificar que el subgrafo de la solución candidata preservara la conectividad y continuara satisfaciendo la demanda de carga entre sus nodos.

De lo anterior surgió la siguiente pregunta: ¿de que manera intentar evitar que el subgrafo se divida en componentes conexos, ya que sólo se requiere tener un único componente conexo después de aplicar el operador de mutación?

Para intentar preservar la conectividad del subgrafo se hizo uso de los puntos de articulación, para influenciar al individuo al ser mutado, ya que en la teoría de grafos estos puntos desempeñan un papel clave para garantizar la conectividad de los mismos. Para la restricción de demanda se consideraron las siguientes preguntas: ¿qué características tienen los nodos que no cumplen con la demanda de carga y por qué se elige cierto nodo para ser eliminado del subgrafo?

Se identificó, que si bien es cierto que se preservaba la conectividad del subgrafo al eliminar un nodo con el costo más grande, existían nodos que no tenían ningún nodo adyacente que estuviera a una distancia menor o igual a αD que cubriera su demanda de carga; por lo que, para eliminar un nodo se verificó que sus nodos adyacentes al menos tuviera un nodo adyacente que sí formara parte de la solución, para que de esta manera se aumentara la posibilidad de satisfacer su demanda de carga.

Después del uso del conocimiento a priori del problema, se decidió utilizar el conocimiento extraído durante la búsqueda realizada por el algoritmo, esta información se tomó del conjunto de individuos aceptados obtenidos por la función de aceptación, en dicho conjunto se identificaron los nodos que con mayor frecuencia eran seleccionados para formar parte de las soluciones candidatas, de tal forma que con esta información se pudiera influir en la selección de los nodos del subgrafo H. Con lo anterior se obtuvo la información necesaria para el uso de una fuente de conocimiento normativo.

Por tanto, se consideraron cuatro tipos de conocimiento: uno situacional, dos de dominio y uno

normativo. Las fuentes de conocimiento que se emplearon en el algoritmo propuesto se listan a continuación:

- Fuente de conocimiento de dominio con costos de construcción.
- Fuente de conocimiento de dominio con puntos de articulación.
- Fuente de conocimiento situacional.
- Fuente de conocimiento normativo.

En la Figura 4.4 se muestra cómo se relacionan las fuentes de conocimiento con las funciones de aceptación, actualización y de influencia en el espacio de creencias.

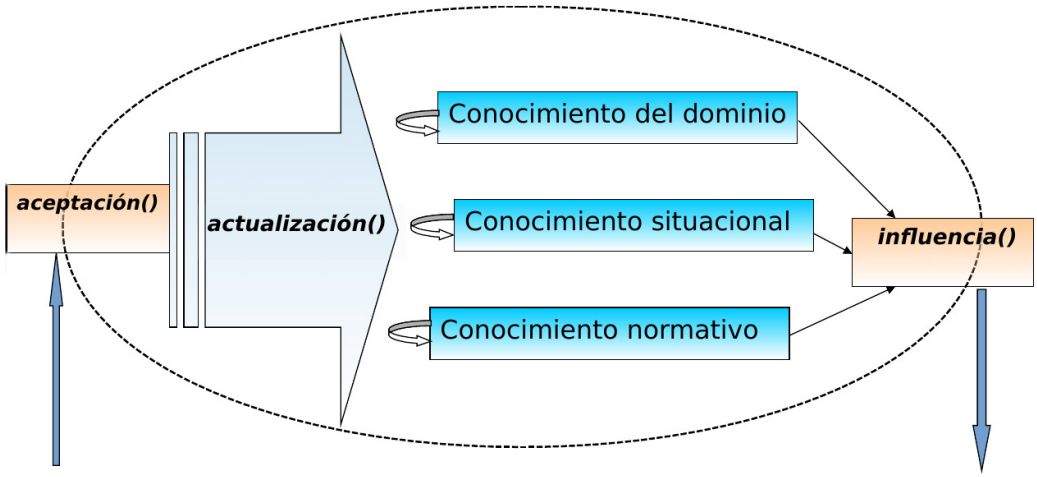


Figura 4.4: Fuentes de conocimiento.

A continuación, se detallan las fuentes de conocimiento empleadas en el diseño del algoritmo cultural propuesto.

4.4.2 Fuente de conocimiento de dominio con costos de construcción.

En esta fuente de conocimiento se aplica información sobre el dominio del problema. En este sentido, dado que se requiere minimizar el costo de construcción de las estaciones de carga, es importante identificar aquel nodo con el costo de construcción más grande para poder eliminarlo, siempre y cuando se conserve la conectividad del subgrafo H de $\hat{G}$ y se tome en cuenta que los nodos adyacentes al nodo a eliminar, tengan a su vez un nodo que pueda cubrir su demanda de carga.

4.4.2.1. Función de influencia

En el Algoritmo 6 se puede apreciar la eliminación del nodo con el costo más grande en la línea 22, donde $x^j_{lc} \leftarrow 0$ indica que el individuo con el índice j en la posición lc se le asigna un valor de cero. Esto siempre y cuando el nodo lc haya cumplido con las condiciones mencionadas en el párrafo anterior. De otro modo, se aplica una mutación uniforme al individuo j con el propósito de mantener la diversidad genética de la población.

La función conexoGrafoHat(x^j) devuelve 0 si no existe penalización en la restricción de conectividad, es decir, que el subgrafo H de $\hat{G}$ es conexo, en otro caso, devuelve un número entero, el cual representa la magnitud de la violación de dicha restricción.

La función demAdyacent(x^j, i) recibe dos parámetros de entrada: el individuo con el índice j y el índice i del nodo candidato a ser eliminado. A partir del nodo i a eliminar se examina cada uno de sus nodos adyacentes k que no pertenecen al conjunto N', es decir, aquellos nodos en x^j con x^j_k=0. Para cada uno de estos nodos examinados k se verifica si tienen al menos un nodo adyacente l que sí pertenezca al conjunto N', es decir, un nodo l con x^j_l=1, con el cual se tenga la posibilidad de cubrir su restricción de demanda, y de este modo poder eliminar al nodo i. La función devuelve 1 si el nodo i tiene un nodo l con x^j_l=1 para cada uno de sus nodos adyacentes k con x^j_k=0. En otro caso, devuelve 0.

Algoritmo 6 Eliminar mayor costo

1: $lc \leftarrow -1$ {Índice del nodo más grande}
2: $largestC \leftarrow 0$ {Costo del nodo más grande}
3: $setN \leftarrow \emptyset$ {Conjunto de nodos a eliminar}
4: **Para** $i \leftarrow 0$ to n **Hacer**
5: **Si** $x_i^j = 1$ **Entonces**
6: $x_i^j \leftarrow 0$
7: **Si** conexoGrafoHat(x^j)=0 and demAdyacent(x^j, i)=1 **Entonces**
8: $setN \leftarrow setN \cup \{i\}$
9: **Fin Si**
10: $x_i^j \leftarrow 1$
11: **Fin Si**
12: **Fin Para**
13: **Para** $i \in setN$ **Hacer**
14: **Si** $c_i > largestC$ **Entonces**
15: $largestC \leftarrow c_i$ {c_i contiene el costo de construcción del nodo i}
16: $lc \leftarrow i$
17: **Fin Si**
18: **Fin Para**
19: **Si** $lc \neq -1$ **Entonces**
20: $x_{lc}^j \leftarrow 0$
21: **Si no**
22: mutacionUniforme(x^j)
23: **Fin Si**

La fuente de conocimiento de dominio no utiliza una función de actualización, debido a que usa conocimiento a priori del dominio del problema, con el cual se conocen qué características deben tener las soluciones, por lo que no se requiere extraer información durante la ejecución del algoritmo. Es por esta razón, que tanto en esta sección como en la siguiente, se omite la subsección *función de actualización*.

4.4.3 Fuente de conocimiento de dominio con puntos de articulación

Otra información inherente al dominio del problema es identificar los puntos de articulación en el grafo que modela una ciudad, con lo cual se intenta mantener una mayor probabilidad de cumplir con la restricción de conectividad del problema del EVCSP. Para lograr esto, se implementó una función llamada articulationPoint(), con la cual se obtienen los puntos de articulación del grafo $\hat{G}$, los cuales se asignan al vector ap (un vector binario de tamaño n). Un ejemplo de puntos de articulación en un detalle de un grafo se muestra en la Figura 4.5.

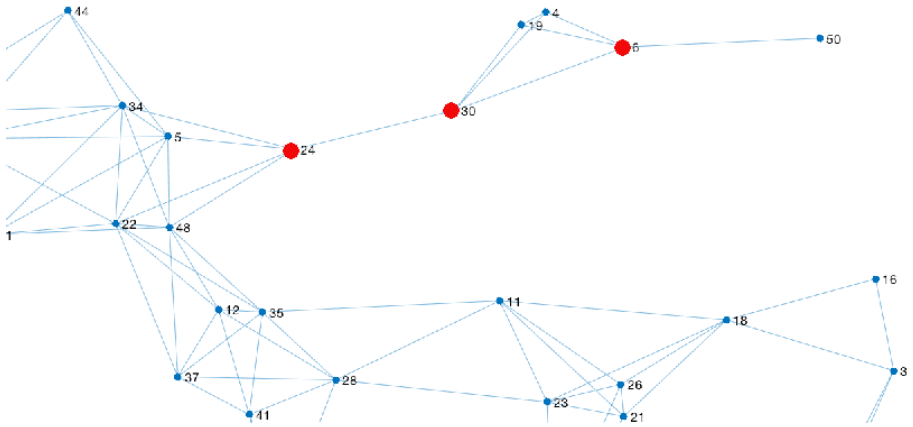


Figura 4.5: Puntos de articulación.

4.4.3.1. Función de influencia

En el Algoritmo 7, en primera instancia se realizó mutación uniforme al individuo x^j, posteriormente se activaron los puntos de articulación cuando se obtienen más de un componente conexo con la función componentesConexos(x^j), la cual devuelve un número entero que representa la cantidad de componentes conexos en el grafo $\hat{G}$. Esto se observa en la línea 5.

Algoritmo 7 Puntos de articulación

```
1: mutacionUniforme(x^j)
2: cc ← componentesConexos(x^j)
3: Si cc > 1 Entonces
4:    Para i ← 1 to n Hacer
5:       Si ap_i = 1 Entonces
6:          x_i ← 1
7:       Fin Si
8:    Fin Para
9: Fin Si
```

4.4.4 Fuente de conocimiento situacional

En este tipo de conocimiento se pretende que los individuos intenten parecerse al mejor individuo hasta el momento.

4.4.4.1. Función de influencia

En la función de influencia del conocimiento situacional, se realizó una mutación con sesgo, en la cual se le asigna a la variable x_i el valor del mejor individuo x^{best} en la posición i. Este procedimiento se realiza con un valor de probabilidad de mutación $pm = 0.02$, el cual es un parámetro que se especifica en la Sección 5.2

En el Algoritmo 8 se muestra la función de influencia del conocimiento situacional.

Algoritmo 8 Conocimiento situacional

1: muestrear $p \sim U(0,1)$
2: **Para** $i \leftarrow 1$ to n **Hacer**
3: **Si** $p \leq pm$ and $x_i^{best} \neq x_i$ **Entonces**
4: $x_i \leftarrow x_i^{best}$
5: **Fin Si**
6: **Fin Para**

4.4.4.2. Función de actualización

En el Algoritmo 9 se muestra cómo se actualiza al mejor individuo encontrado hasta la generación anterior x^{best} con el valor del nuevo mejor individuo recién encontrado en la generación actual $x^{newBest}$, siempre que la magnitud de penalización de éste en $\phi(x^{newBest})$ sea menor o igual a la penalización del mejor individuo encontrado hasta la generación anterior $\phi(x^{best})$. Esto se cumple cuando en las primeras generaciones aún se obtienen soluciones no factibles. En el tiempo que el algoritmo ya obtiene soluciones factibles, la actualización del mejor individuo se realiza cuando el valor de la función objetivo del nuevo mejor individuo $f(x^{newBest})$ es menor o igual al del valor $f(x^{best})$ del mejor individuo encontrado. En otro caso, se actualiza al peor individuo $x^{secondBest}$ con el valor del mejor individuo encontrado.

4.4.5 Fuente de conocimiento normativo

Dado el conjunto de individuos aceptados de acuerdo con la función de aceptación (ver Sección 4.4.6), se obtuvieron frecuencias por cada una de sus variables x_i en el caso de que éstas sean seleccionadas como ubicaciones para construir estaciones de carga, es decir, las variables $x_i = 1$. Dichas frecuencias sirven como guía para modificar al operador de mutación.

Algoritmo 9 Actualización del mejor individuo

Si $\phi(x^{newBest}) \leq \phi(x^{best})$ and $\phi(x^{newBest}) \neq 0$ **Entonces**
  $x^{best} \leftarrow x^{newBest}$
  $\phi(x^{best}) \leftarrow \phi(x^{newBest})$
  $f(x^{best}) \leftarrow f(x^{newBest})$
Fin Si
Si $\phi(x^{newBest}) = 0$ and **(**$\phi(x^{best}) > 0$ or $f(x^{newBest}) \leq f(x^{best})$**) Entonces**
  $x^{best} \leftarrow x^{newBest}$
  $\phi(x^{best}) \leftarrow \phi(x^{newBest})$
  $f(x^{best}) \leftarrow f(x^{newBest})$
Si no
  $x^{secondBest} \leftarrow x^{best}$
  $\phi(x^{secondBest}) \leftarrow \phi(x^{best})$
  $f(x^{secondBest}) \leftarrow f(x^{best})$
Fin Si

4.4.5.1. Función de influencia

En el Algoritmo 10 se observa cómo se pretende seleccionar, para la construcción, a aquellos nodos que vayan con la tendencia del conjunto de individuos aceptados, donde $|accept|$ es el número de individuos aceptados de acuerdo con la Ecuación 4.6, que se describirá más adelante, y BC es el vector que contiene las frecuencias de las variables x_i del conjunto de individuos aceptados.

Algoritmo 10 Conocimiento normativo

muestrear $p \sim U(0,1)$
Para $i \leftarrow 1$ to n **Hacer**
  Si $p \leq \frac{BC_i}{|accept|}$ **Entonces**
    $x_i \leftarrow 1$
  Fin Si
Fin Para

4.4.5.2. Función de actualización

En esta función de actualización del conocimiento normativo, se actualiza al conjunto de individuos aceptados, sólo si la suma de sus promedios tanto de los valores de la función objetivo $avgfAccept$, como de las penalizaciones $avg\phi Accept$ es mayor que la suma de los promedios de los

individuos aceptados de la población actual $(avgf + avg\phi)$. Así también, se actualizan las frecuencias en el vector BC.

En el Algoritmo 11 se observa en las líneas de la 1 a la 8 cómo se obtienen los promedios tanto de los valores de la función objetivo $avgf$, como de las penalizaciones de los individuos aceptados $avg\phi$ de la población actual. Esto se calcula con el propósito de actualizar al conjunto de individuos aceptados $accept$ con los individuos aceptados de la población actual P. Una vez hecho esto, se obtienen las frecuencias de cada una de las variables $accept_i^j = 1$ en el vector BC, en la línea 14. Por último, en las líneas 18 y 19 se actualizan los promedios de los valores de la función objetivo $avgfAccept$ y de las penalizaciones $avg\phi Accept$ con los promedios de los individuos aceptados de la población actual.

Algoritmo 11 Actualización del conocimiento normativo

1: $sumf \leftarrow 0$
2: $sum\phi \leftarrow 0$
3: **Para** j to $|accept|$ **Hacer**
4: $\quad sumf \leftarrow sumf + f(x^j)$
5: $\quad sum\phi \leftarrow sum\phi + \phi(x^j)$
6: **Fin Para**
7: $avgf \leftarrow \frac{sumf}{|accept|}$ {Promedio de los valores de la función objetivo de los individuos aceptados de la población actual}
8: $avg\phi \leftarrow \frac{sum\phi}{|accept|}$ {Promedio de las penalizaciones de los individuos aceptados de la población actual}
9: **Si** $(avgf + avg\phi) < (avgfAccept + avg\phi Accept)$ **Entonces**
10: $\quad$ **Para** $j \in accept$ **Hacer**
11: $\quad\quad$ **Para** $i \leftarrow 1$ to n **Hacer**
12: $\quad\quad\quad accept_i^j \leftarrow accept_i^j \cup x_i^j$
13: $\quad\quad\quad$ **Si** $accept_i^j = 1$ **Entonces**
14: $\quad\quad\quad\quad BC_i \leftarrow BC_i + 1$
15: $\quad\quad\quad$ **Fin Si**
16: $\quad\quad$ **Fin Para**
17: $\quad$ **Fin Para**
18: $\quad avgfAccept \leftarrow avgf$
19: $\quad avg\phi Accept \leftarrow avg\phi$
20: **Fin Si**

4.4.6 Función de aceptación

La función de aceptación determina cuáles individuos y sus comportamientos pueden impactar el conocimiento del espacio de creencias. En la Ecuación 4.6 se muestra la función de aceptación dinámica empleada en el algoritmo propuesto, la cual cambia el número de individuos aceptados en cada generación, es decir, que en la generación número uno, la cantidad de individuos aceptados se duplica, y conforme el número de generaciones incrementa, el número de individuos aceptados decrece.

$$accept = top\left(p\,\% + \left\lfloor \frac{p\,\%}{k} \right\rfloor\right) \tag{4.6}$$

donde $p\,\%$ representa el porcentaje mínimo permitido del espacio de la población que afectará el espacio de creencias. El valor que se usó para $p\,\%$ fue del $20\,\%$ de la población de 30 individuos, ya que se recomiendan valores menores a 50 %, y de preferencia alrededor de 10 %, de acuerdo con el trabajo de Landa [23] y adicionalmente 20 % corresponde con la "la regla del éxito $1/5$" [49]. k es el número de generaciones y $top(m)$ es una función que selecciona los el m % mejores individuos.

4.4.7 Función de influencia principal

La función de influencia principal se realizó mediante selección por ruleta de acuerdo con Iacoban et al. [15]. En la Ecuación 4.7 se obtiene el promedio del valor de aptitud de los individuos generados usando la función de influencia q.

$$avr_q = \frac{\sum_{j \in O_q} fit(x^j)}{|O|} \tag{4.7}$$

donde O_q es el conjunto de individuos generados por la función de influencia q; $fit(x^j)$ es el valor de aptitud del individuo j. Posteriormente, al operador de influencia se le asigna un área en la rueda de ruleta relativa al promedio de su desempeño de acuerdo a la Ecuación 4.8.

$$p_q = \frac{avr_q}{\sum_{j=1}^{nOp} avr_j} \tag{4.8}$$

donde p_q es un porcentaje asignado a la rueda de ruleta para influenciar al operador q; y nOp es el número de operadores de influencia utilizados (4 en este caso).

Para el algoritmo cultural propuesto, se emplearon 4 funciones de influencia, las cuales fueron descritas en las secciones anteriores. Para inicializar la rueda de ruleta, se asignó un área igual para cada tipo de función de influencia, quedando cada una con un porcentaje del 25 %. En caso de que no se cuente con ningún individuo generado mediante alguna función de influencia, se asigna un límite inferior de 0.1, para garantizar una probabilidad mayor que cero asignada a la rueda de ruleta para todas las funciones de influencia.

4.5 Resumen del capítulo

En este capítulo se describieron los pasos inherentes al desarrollo de la metodología, así como el manejo de las restricciones del EVCSP empleando el enfoque de jerarquías estocásticas. Por otro lado, se detallaron tanto la representación y operadores evolutivos del algoritmo genético, como cada uno de los componentes del algoritmo cultural propuesto, incluyendo a las cuatro fuentes de conocimiento diseñadas; dos fuentes de conocimiento de dominio, una con respecto a los costos de construcción y la otra a los puntos de articulación, la tercera fuente de conocimiento fue la situacional y la cuarta la de conocimiento normativo. Por último, se describieron las funciones de aceptación y de influencia que representan el protocolo de comunicación para un algoritmo cultural.

5

Experimentación y resultados

En este capítulo se describen, inicialmente, las instancias del problema y la configuración empleada para la experimentación; posteriormente, se describen las experimentaciones y resultados obtenidos con cada uno de los algoritmos probados: el algoritmo genético simple, el algoritmo cultural propuesto y el algoritmo voraz del estado del arte.

5.1 Descripción de las instancias del problema

Las instancias del problema con las que se realizaron las experimentaciones son de tamaños 10, 50, 100, 150 y 200 nodos, cada una con diferente número de aristas. La información con la que cuenta cada instancia se lista a continuación:

- Número de nodos del grafo.
- Número de aristas del grafo.
- Distancia de la autonomía del EV.

- Aristas del grafo (tuplas).
- Coordenadas de cada nodo.
- Costo de construcción de cada estación de carga.
- Capacidad de carga de cada estación de carga.
- Requerimiento de demanda de cada estación de carga.

Los parámetros de configuración que se usaron para realizar la experimentación se muestran en la Tabla 5.1.

Parámetro	Descripción	Valor
D	Autonomía del EV	40 km para 10 nodos, 20 km para 50 nodos, 15 km para 100, 150 y 200 nodos
f_i	Capacidad de carga de la estación de carga	0.5 para cada nodo i
F_i	Requerimiento de demanda de la estación de carga	1.0 para cada nodo i
c_i	Costo de construcción	un valor aleatorio en el rango de (0,1] para cada nodo i
α	Factor de descuento	1

Tabla 5.1: Configuración de la simulación

Para la generación de cada instancia del grafo G, digamos con una instancia de tamaño 50, se ubican aleatoriamente 50 nodos en un área de 100 $\times$ 100 km^2, donde al costo de c_i se le asigna un valor aleatorio en el rango de (0,1] y a D, f_i y F_i, para cada i, se les asignan 20 km, 0.5, y 1, respectivamente. Por simplicidad, se asume que los nodos están interconectados y que la longitud de la ruta más corta de cada par de nodos está determinada con la distancia euclidiana entre ellos.

Cabe mencionar, que las instancias empleadas para la experimentación fueron las que se usaron en el trabajo de Lam et al. [22], siendo ellos quienes amablemente nos las compartieron.

5.2 Experimentación

Inicialmente se realizó experimentación con un algoritmo genético simple con los operadores que se especificaron en la Sección 4.3; con esta experimentación se pudo observar el comportamiento del problema del EVCSP al ser abordado desde el enfoque del cómputo evolutivo. Posteriormente, se realizó experimentación con el algoritmo cultural propuesto y se compararon los resultados con el algoritmo voraz, el cual se implementó de acuerdo con el trabajo de Lam et al. [22]. En dicho trabajo no se muestran los resultados del Método I y el Método III para las instancias de tamaño 200 ya que estos no son computables por YALMIP/CPLEX debido al problema de memoria insuficiente. Lo cual implica que los enfoques MILP no son adecuados para problemas grandes.

Los tamaños de las instancias con las que se llevó a cabo la experimentación son de 10, 50, 100, 150 y 200 nodos. Para cada uno de los 5 tamaños se utilizaron 100 instancias sintéticas.

Para el caso del algoritmo genético simple y el algoritmo cultural propuesto, se realizaron 30 ejecuciones por instancia. Por lo contrario, se realizó sólo una ejecución por instancia en el caso del algoritmo voraz del estado del arte, debido a que éste es determinista. Los parámetros del algoritmo cultural propuesto se muestran en la Tabla 5.2, dichos parámetros se utilizaron de acuerdo con los sugeridos en la literatura [40], el número de generaciones se definió en función del número de nodos, ya que para instancias pequeñas de 10 y 50 nodos, los algoritmos convergen en pocas generaciones, y para instancias grandes se requiere más generaciones para obtener mejores soluciones.

Parámetro	Valor
Número de generaciones	10n
Tamaño de la población	30
Probabilidad de cruza	0.7
Probabilidad de mutación	0.02

Tabla 5.2: Parámetros del algoritmo genético y del algoritmo cultural propuesto

5.3 Resultados obtenidos empleando el algoritmo genético simple

Con el fin de evidenciar la cantidad de resultados que mejoran al algoritmo voraz empleando el algoritmo genético usando las instancias de tamaño 100, en la Tabla 5.3 se muestra la comparación de los resultados entre los algoritmos voraz y genético simple con las 100 instancias de tamaño 100. Ya que para las 100 instancias tanto de tamaño 10 como de 50, el algoritmo genético mejora en todos los resultados al algoritmo voraz.

En la columna "Inst" se encuentra el número de la instancia, en la columna "AV" el costo de construcción empleando el algoritmo voraz, en las columnas "Promedio", "Mediana" y"Mejor" están dichas estadísticas de los costos de construcción de las 30 ejecuciones que se realizaron por instancia y en la columna "EC" el número de estaciones de carga obtenido empleando el algoritmo genético simple "AG".

Se puede observar que el algoritmo genético mejora el costo de construcción del algoritmo voraz sólo en 24 de las 100 instancias; esto indica que para 100 nodos el algoritmo genético no obtiene mejores resultados que el voraz.

Se calcularon los promedios de los resultados obtenidos con las 100 instancias por cada uno de los 5 tamaños, y se observó que el algoritmo genético mejora en promedio los costos de construcción del

algoritmo voraz en instancias con tamaño 10 y 50; sin embargo, a partir de 100 nodos el algoritmo genético no es capaz de mejorar al voraz. Esto se puede observar en la Tabla 5.4, donde se resaltan los mejores resultados en negritas y la columna "No.Nodos" indica el tamaño de las instancias con las que se experimentó, las columnas llamadas "Prom. Mejores" indican los promedios de los mejores costos de construcción de las 100 instancias por cada tamaño, en la columna "Gran Prom." se muestra el gran promedio (el promedio de los promedios de las 30 ejecuciones que se realizaron por instancia), en la columna "No. Éxitos" se indica el número de ejecuciones en las que el algoritmo genético le gana al voraz.

Es primordial mencionar que la implementación de un algoritmo genético simple se justifica, por una parte como un punto de referencia para iniciar el diseño del algoritmo propuesto en esta tesis, por otra parte es importante resaltar que ante la falta de un algoritmo genético en la literatura para realizar las comparaciones con el método propuesto, se tomó la decisión de implementar una metaheurística genérica, de tal manera que al no ser este el objetivo principal de la tesis, no se incorporaron operadores o algún tipo de conocimiento del dominio del problema para que el algoritmo genético tuviera un mejor desempeño que el algoritmo voraz del estado del arte.

Lo anterior, da la pauta para pensar que el algoritmo voraz del estado del arte al basarse en el conocimiento específico del problema para intentar resolverlo, generar soluciones que satisfacen las restricciones del mismo, obteniendo de esta manera mejores soluciones que el algoritmo genético implementado.

Inst.	AV	AG				Inst.	AV	AG			
		Promedio	Mediana	Mejor	EC			Promedio	Mediana	Mejor	EC
1	13.143	15.428	15.595	**12.865**	35	51	14.347	17.163	16.987	14.731	37
2	9.714	11.712	11.663	10.291	32	52	15.496	18.026	17.896	15.682	38
3	12.797	15.388	15.403	13.829	34	53	11.817	14.182	14.018	12.526	35
4	15.134	17.073	17.040	15.138	35	54	11.221	13.111	13.015	11.467	32
5	10.096	12.559	12.485	10.932	31	55	9.197	11.399	11.259	9.950	33
6	10.853	13.119	12.995	11.470	31	56	14.725	17.440	17.452	15.953	35
7	10.694	13.302	13.400	10.949	33	57	12.922	14.263	14.208	**12.417**	36
8	13.093	14.650	14.584	**13.020**	34	58	10.94	13.288	13.308	11.813	34
9	12.706	15.525	15.679	13.994	35	59	11.123	13.437	13.130	11.466	32
10	10.814	12.449	12.405	**10.715**	33	60	11.992	14.284	14.204	12.073	33
11	11.738	13.603	13.304	12.102	33	61	13.993	15.292	15.207	**13.866**	34
12	10.964	13.265	13.036	12.141	33	62	13.725	16.029	16.049	14.221	38
13	11.387	14.344	14.173	12.505	36	63	10.684	13.974	13.819	12.113	34
14	13.105	15.021	14.929	13.440	34	64	12.565	13.948	13.654	12.684	30
15	14.461	15.590	15.355	**14.141**	35	65	13.846	16.232	16.122	14.835	32
16	11.772	14.420	14.148	12.515	34	66	10.116	12.614	12.557	10.939	32
17	12.188	13.426	13.197	**11.901**	32	67	13.265	14.770	14.692	13.513	35
18	13.601	15.118	15.195	**13.500**	33	68	10.666	12.159	12.167	10.904	29
19	14.379	15.097	15.194	**13.197**	33	69	13.059	15.669	15.596	13.955	33
20	9.999	11.991	12.067	10.529	32	70	12.443	14.599	14.394	12.695	31
21	8.093	11.049	11.085	9.219	31	71	10.912	13.794	13.943	11.828	37
22	9.890	12.772	12.761	11.076	35	72	11.612	14.277	14.183	**11.537**	34
23	12.441	14.157	14.077	**12.356**	35	73	12.373	14.959	14.875	13.032	32
24	11.008	12.362	12.248	**10.671**	33	74	9.352	11.626	11.534	9.716	31
25	7.792	10.074	10.019	8.311	30	75	15.551	17.436	17.390	**14.858**	35
26	12.783	14.853	14.840	**12.482**	31	76	11.03	13.010	13.133	11.779	30
27	10.955	13.482	13.203	11.696	32	77	10.293	11.733	11.580	10.616	32
28	10.432	12.036	12.026	10.438	35	78	11.569	15.222	15.127	12.398	34
29	11.15	12.483	12.274	**10.669**	31	79	10.557	12.729	12.593	**10.260**	30
30	10.631	13.260	13.277	11.529	32	80	12.502	15.281	14.977	13.333	38
31	14.99	16.540	16.537	**14.606**	32	81	10.859	13.035	12.818	11.036	34
32	9.034	11.972	11.989	10.025	32	82	12.794	15.490	15.192	13.261	36
33	11.107	13.777	13.602	11.789	35	83	12.426	14.837	15.079	12.517	38
34	14.782	18.023	17.764	16.410	31	84	13.553	15.668	15.542	13.891	34
35	14.593	16.349	16.073	**13.965**	36	85	10.757	13.195	13.149	11.449	33
36	12.256	14.665	14.632	13.108	37	86	13.835	15.850	15.644	13.969	36
37	12.047	13.922	13.717	12.884	33	87	11.894	13.827	13.793	12.260	32
38	14.752	16.967	16.833	15.618	36	88	11.101	13.455	13.402	11.745	33
39	15.66	18.386	18.264	16.567	35	89	10.774	12.478	12.389	**10.544**	29
40	10.992	13.833	13.742	12.338	36	90	13.129	15.556	15.403	14.184	36
41	11.281	13.676	13.534	12.446	31	91	10.696	12.312	12.406	11.052	32
42	15.094	16.787	16.736	**14.156**	32	92	11.842	14.202	14.241	12.478	32
43	12.664	14.686	14.424	13.180	33	93	12.927	14.954	14.932	13.180	35
44	15.067	17.061	16.852	15.505	36	94	14.397	15.606	15.600	**13.850**	33
45	12.007	14.355	14.259	12.007	32	95	12.812	14.024	14.056	**12.041**	32
46	9.431	11.363	11.325	9.694	32	96	11.945	14.452	14.422	12.117	34
47	11.569	14.468	14.440	12.259	35	97	12.059	14.413	14.446	12.525	34
48	11.761	13.717	13.805	12.419	36	98	13.48	13.944	13.883	**12.456**	30
49	9.763	11.969	12.005	**9.631**	30	99	11.192	13.868	13.822	12.267	30
50	13.374	15.880	15.931	13.932	32	100	10.007	12.438	12.507	10.100	32

Tabla 5.3: Comparación de los resultados entre los algoritmos voraz y genético simple con las 100 instancias de tamaño 100, donde los valores en negritas en la columna "Mejor" son los que mejoran al algoritmo voraz.

No. Nodos	AV	AG		
	Prom. Mejores	Gran Prom.	Prom. Mejores	No. Éxitos
10	1.600	1.599	**1.562**	2679
50	7.353	7.717	**7.151**	719
100	**12.064**	14.236	12.442	53
150	**9.372**	13.379	11.169	0
200	**7.316**	12.490	10.173	0

Tabla 5.4: Comparación de los promedios de las 100 instancias por cada tamaño entre los algoritmos voraz y genético, donde se resaltan los mejores resultados en negritas.

5.4 Comparación de los resultados

A continuación, se muestra la comparación de los resultados del algoritmo propuesto con los obtenidos empleando los algoritmos genético y voraz.

En la Tabla 5.5 la columna "No.Nodos" indica el tamaño de las instancias con las que se experimentó, las columnas llamadas "Prom. Mejores" indican los promedios de los mejores costos de construcción de las 100 instancias por cada tamaño, obtenidos por los algoritmos voraz, genético y cultural, respectivamente, en las columnas "Gran Prom." se muestra el gran promedio (el promedio de los promedios de las 30 ejecuciones que se realizaron por instancia) de los algoritmos genético y cultural, en la columna "No. Éxitos" se indica el número de ejecuciones en las que el algoritmo cultural le gana al voraz.. Finalmente en las columnas "Tiempo en seg." y "No. EC" se observan los promedios de los tiempos de ejecución y de estaciones de carga tanto del algoritmo voraz como del cultural.

Para observar el detalle de los resultados obtenidos empleando los algoritmos voraz y cultural con las 100 instancias por cada uno de los cinco tamaños, se tienen las Tabla A.1, A.2, A.3, A.4 y

A.5 del anexo A, donde en la última fila se muestran los promedios de los costos de construcción, los cuales son los mismos que aparecen en la Tabla 5.5.

Los valores en negritas indican aquellos promedios que mejoraron al algoritmo voraz del estado del arte. Como se puede observar, el algoritmo cultural propuesto obtiene valores competitivos con respecto al voraz; sin embargo, en cuanto al tiempo de ejecución, el algoritmo cultural ocupa mayor tiempo, debido al número de instrucciones ejecutadas por generación.

No. Nodos	AV	AG		AC			Tiempo en seg.		No. EC	
	Prom. Mejores	Gran Prom.	Prom. Mejores	Gran Prom.	Prom. Mejores	No. Éxitos	AV	AC	AV	AC
10	1.600	1.599	1.562	1.564	**1.562**	2983	0.000	0.003	3.800	3.650
50	7.353	7.717	7.151	7.371	**7.131**	1978	0.006	0.174	19.250	18.460
100	12.064	14.236	12.442	12.163	**11.545**	1465	0.040	2.459	35.270	33.450
150	9.372	13.379	11.169	9.770	**8.933**	920	0.240	8.840	38.520	35.980
200	7.316	12.490	10.173	7.901	**7.065**	511	0.805	20.697	39.800	36.760

Tabla 5.5: Estadísticas de los promedios de los valores obtenidos de las 100 instancias por cada tamaño, donde se resaltan los mejores resultados en negritas.

5.5 Análisis del desempeño del algoritmo cultural propuesto y de las fuentes de conocimiento

A continuación, se muestran los resultados del algoritmo cultural propuesto y la comparación del mismo con los algoritmos genético y voraz.

5.5.1 Gráficas de convergencia

En la Figura 5.1 se muestra la convergencia tanto del algoritmo genético como del algoritmo cultural propuesto. Tal gráfica exhibe el comportamiento de la ejecución en la mediana de las 30 ejecuciones y empleando las instancias de tamaño 10. En esta gráfica se observa cómo el algoritmo cultural AC tiene una convergencia más rápida que el algoritmo genético AG.

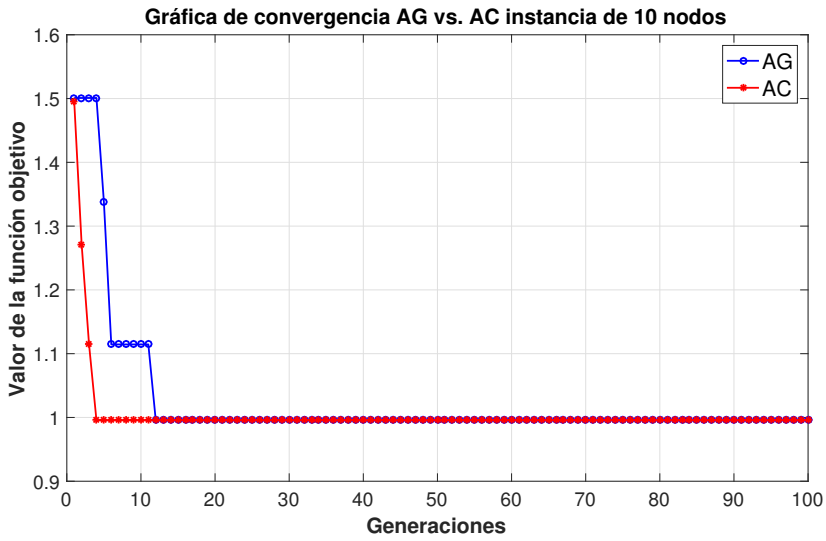


Figura 5.1: Gráfica de convergencia de los algoritmos genético y cultural propuesto empleando la instancia de 10 nodos.

En las Figuras 5.2, 5.3 y 5.4 se muestra la convergencia entre los algoritmos genético AG y cultural AC con diferentes tamaños de problemas, donde se observa cómo el segundo tiene una convergencia más veloz que el primero.

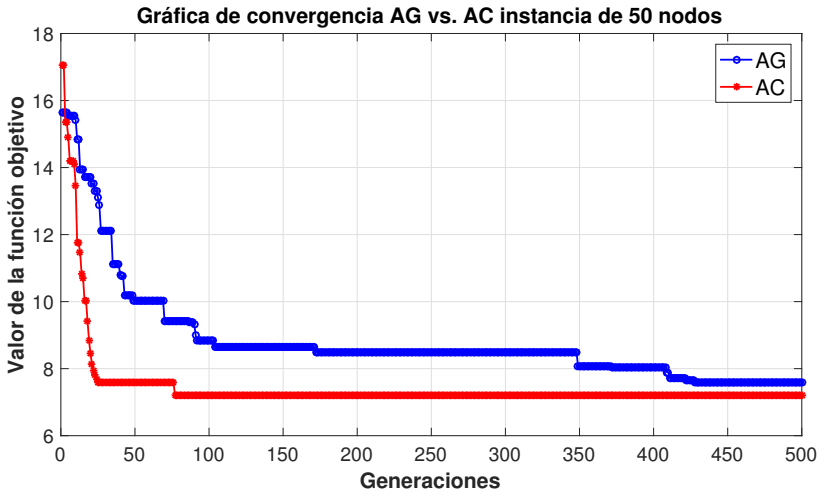


Figura 5.2: Gráfica de convergencia de los algoritmos genético y cultural propuesto empleando la instancia de 50 nodos.

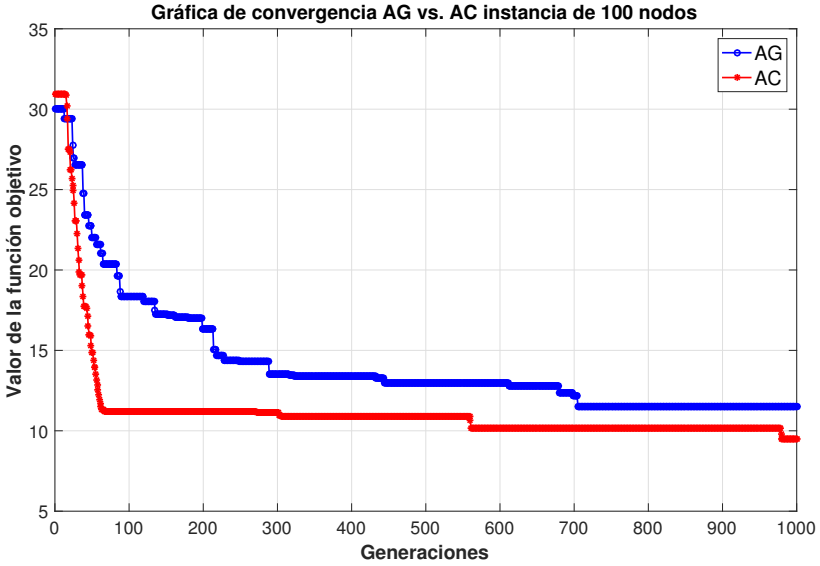


Figura 5.3: Gráfica de convergencia de los algoritmos genético y cultural propuesto empleando la instancia de 100 nodos.

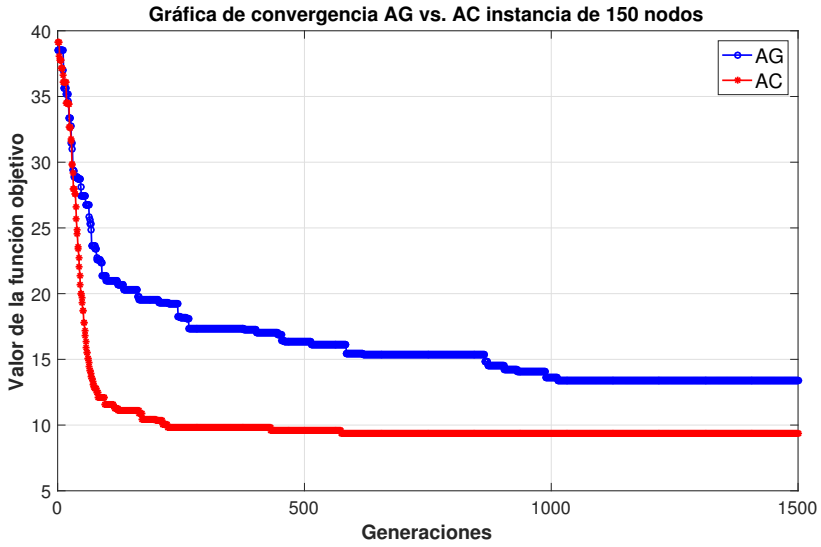


Figura 5.4: Gráfica de convergencia de los algoritmos genético y cultural propuesto empleando la instancia de 150 nodos.

En la Figura 5.5 se muestra la convergencia entre los algoritmos genético AG y cultural AC empleando la instancia con 200 nodos, se puede observar cómo, antes de las 50 generaciones, el algoritmo cultural ha alcanzado una mejor solución que el algoritmo genético al terminar las 2000 generaciones. Esto indica que el algoritmo cultural propuesto explora rápidamente las regiones promisorias y logra explotarlas adecuadamente.

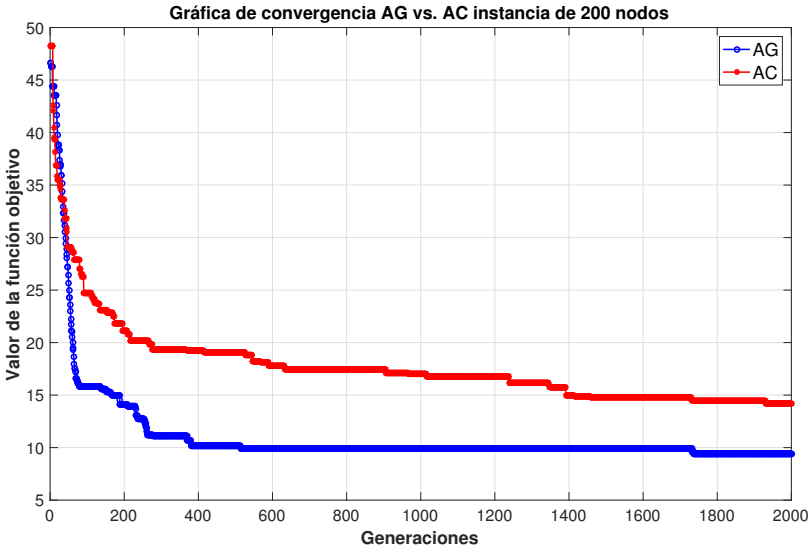


Figura 5.5: Gráfica de convergencia de los algoritmos genético y cultural propuesto empleando la instancia de 200 nodos.

En los experimentos realizados con ambos algoritmos, tanto genético como cultural, se identificó que el algoritmo cultural obtuvo una mejora considerable con respecto al genético simple, ya que fue capaz de encontrar mejores soluciones utilizando los operadores con influencia cultural. Las gráficas de convergencia anteriores mostraron que la incorporación de conocimiento cultural aceleró la convergencia del algoritmo propuesto. De la misma manera se pueden corroborar los resultados en la Tabla 5.5.

Por otra parte, es importante mencionar que al definir el número de generaciones proporcional al tamaño del problema, ambos algoritmos lograron alcanzar mejores soluciones principalmente con las

instancias con mayor número de nodos; sin embargo, se observó que en las últimas etapas del proceso de búsqueda la velocidad de convergencia fue muy lenta estancándose en varias generaciones. Lo anterior, da la pauta a pensar que para evitar que el algoritmo se estanque en óptimos locales, se podría mejorar la presión selectiva. Además, se podría mejorar la capacidad de mostrar diferentes partes del espacio de búsqueda con la adecuación de los operadores utilizados.

5.5.2 Contribución de las fuentes de conocimiento

A continuación, se muestran las gráficas de casos de éxito de cada una de las fuentes de conocimiento que componen el diseño del algoritmo cultural propuesto. En dichas gráficas se observa la frecuencia en que cada una de las fuentes de conocimiento mejoran la aptitud de los individuos mutados en cada generación.

Con respecto a las Figuras 5.6, 5.7 y 5.8, de las instancias de 10, 50 y 100 nodos respectivamente; se observa que la fuente de conocimiento de dominio con costos obtiene un mayor número de casos de éxito en las primeras generaciones, mientras que la fuente de conocimiento normativo mantiene un número de casos de éxito más o menos constante en las generaciones subsecuentes. Sin embargo, son mayores los casos de éxito de la fuente de conocimiento de dominio con costos, con respecto a las otras tres fuentes de conocimiento. Esto es porque esta fuente de conocimiento contiene información con la cual es más probable generar soluciones factibles, ya que se usa conocimiento específico del problema del EVCSP. En este caso, para poder satisfacer la restricción de conectividad del problema, se toma en cuenta la identificación de aquellos nodos i con x_i=1 con los cuales se mantenga conexo al subgrafo inducido H de $\hat{G}$, aun cuando algún nodo i se elimine de H, esto es, poner x_i=0; y para tratar de cubrir la restricción de demanda, se considera la verificación de que los nodos adyacentes al nodo a eliminar x_i=0, tengan un nodo que pueda cubrir su demanda de carga.

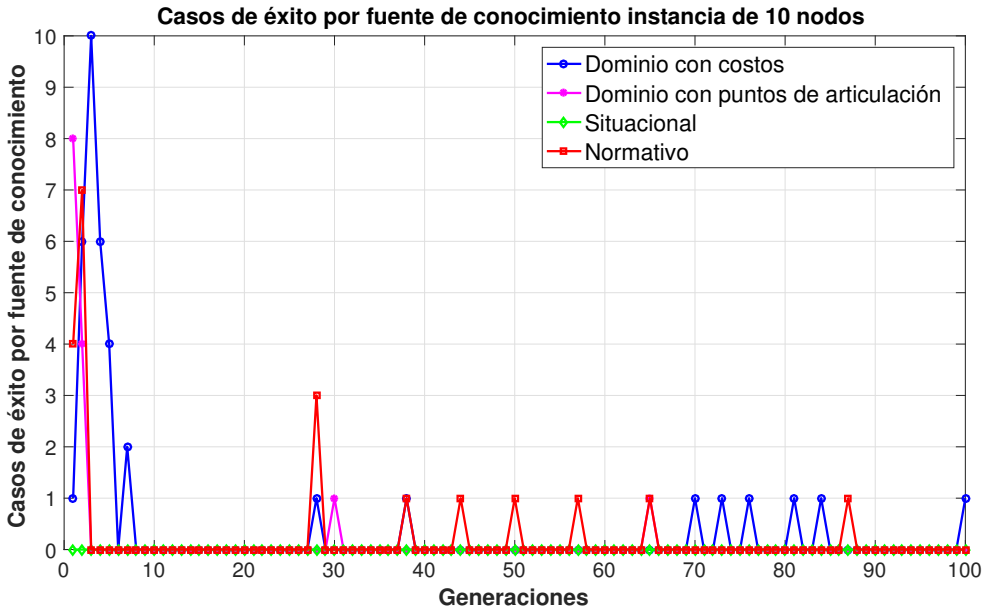


Figura 5.6: Gráfica de casos de éxito con la instancia de 10 nodos.

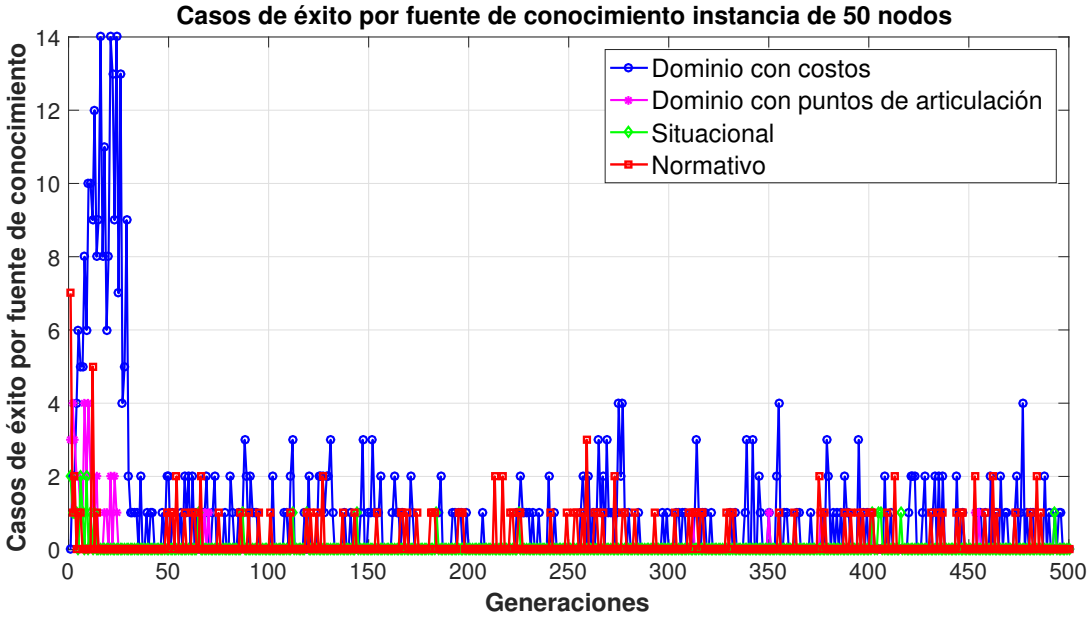


Figura 5.7: Gráfica de casos de éxito con la instancia de 50 nodos.

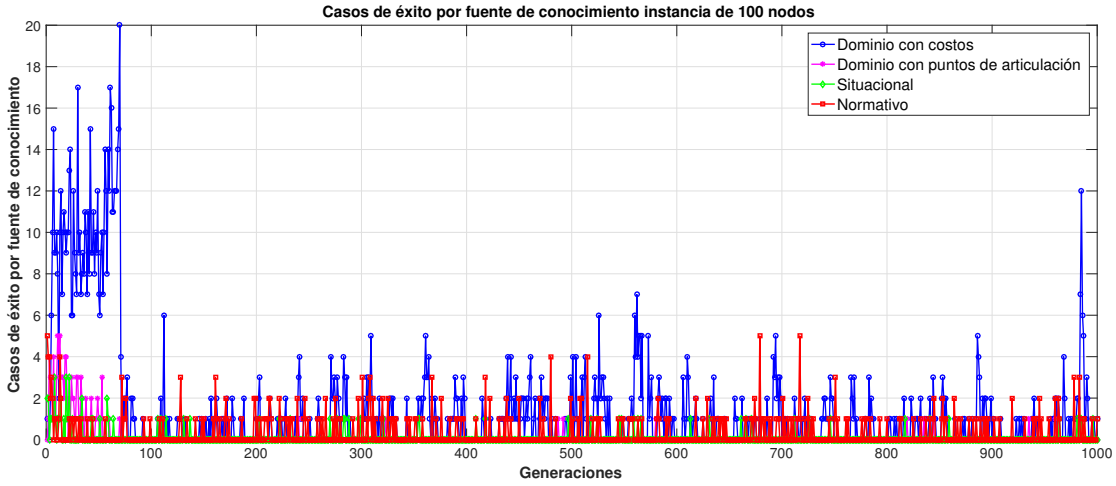


Figura 5.8: Gráfica de casos de éxito con la instancia de 100 nodos.

En la Figura 5.9 para la instancia con 150 nodos, se observa que la fuente de conocimiento normativo obtiene mayor número de casos de éxito con respecto a las fuentes de conocimiento de dominio con puntos de articulación y situacional; sin embargo, la fuente con mayor tasa de casos de éxito es la fuente de conocimiento de dominio con costos. La fuente de dominio con puntos de articulación, si bien ocupa conocimiento inherente al dominio del problema, lo cual ayuda en el proceso de búsqueda, en este caso, obtiene un menor número de casos de éxito que la fuente de dominio con costos, debido a que sólo trata de satisfacer la restricción de conectividad.

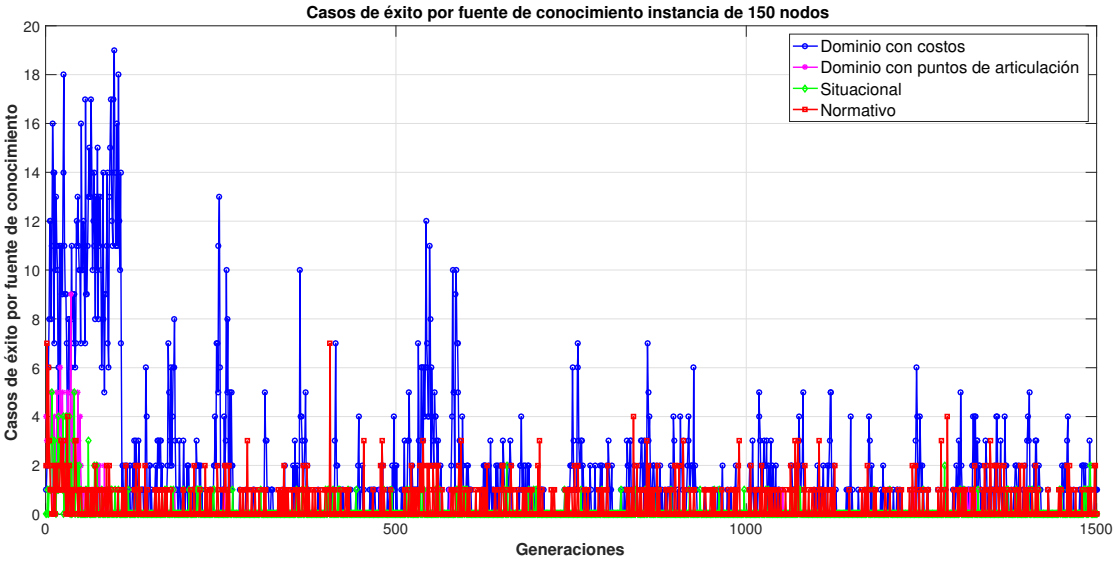


Figura 5.9: Gráfica de casos de éxito con la instancia de 150 nodos.

En la Figura 5.10 con la instancia de 200 nodos se observa que la fuente de conocimiento de dominio con costo supera a las otras tres fuentes de conocimiento. Por otro lado, se aprecia que las fuentes de conocimiento de dominio con puntos de articulación y situacional, tienen más casos de éxito sólo al inicio de las generaciones, mientras que la fuente de conocimiento normativo aporta casos de éxito a lo largo del proceso evolutivo.

Como se mencionó anteriormente la fuente de conocimiento de dominio con costos de construcción, usa conocimiento a priori del dominio del problema, a diferencia de la fuente de conocimiento normativo, que extrae información durante el proceso evolutivo para basarse en la experiencia de los individuos que usaron la fuente de conocimiento con anterioridad. En este sentido, este conocimiento aporta mejoras en la aptitud de los individuos en cada generación, ya que se basa en la selección de aquellos nodos que aparezcan con mayor frecuencia en el conjunto de individuos aceptados, siempre que el promedio de su aptitud mejore al obtenido en la generación anterior.

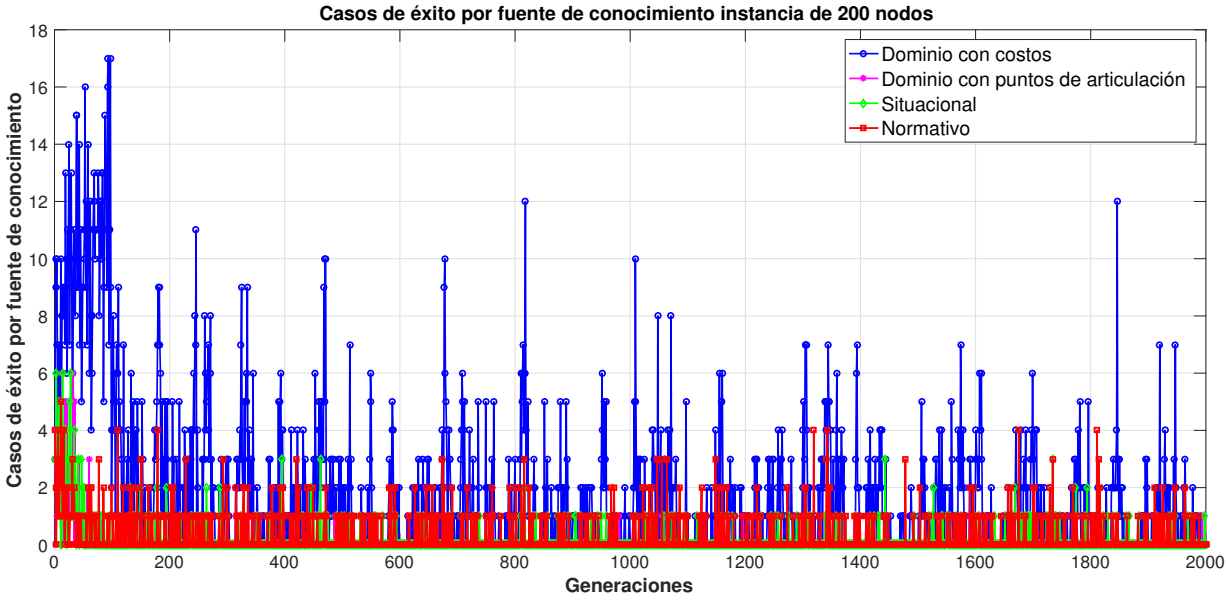


Figura 5.10: Gráfica de casos de éxito con la instancia de 200 nodos.

5.6 Análisis de significación estadística

Con el fin de comparar el comportamiento de los algoritmos evolutivos (genético y cultural), se realizó el análisis estadístico de los resultados obtenidos. Dado que una de las condiciones que se necesita satisfacer para el uso seguro de las pruebas paramétricas es la normalidad, inicialmente se realizó una prueba de *Kolmogorov-Smirnov* para indicar la presencia o ausencia de esta condición en los resultados. Como la condición de normalidad sólo se cumplió para los resultados obtenidos usando las instancias de tamaño 150 y 200, y para algunos de los resultados utilizando las instancias de los tamaños 10, 50 y 100, se decidió utilizar una prueba no paramétrica en todos los casos para mantener la uniformidad, siendo la prueba de *Wilcoxon* la elegida con un nivel de significación $\alpha = 0.05$.

Para los resultados obtenidos empleando las instancias de tamaño 10, sólo 28 de las 100 instancias tuvieron diferencia de significación a favor del algoritmo cultural y con las instancias de tamaño 50, sólo 89, el resto de los resultados quedaron sin diferencia significativa entre los algoritmos comparados.

Por otra parte, para los resultados utilizando las instancias de mayor tamaño, es decir, para los tamaños de 100, 150 y 200 nodos, se obtuvieron diferencias significativas a favor del algoritmo cultural, donde la prueba de *Wilcoxon* obtiene valores p menores que el nivel de significación $\alpha = 0.05$.

En la Tabla 5.7 se muestra en la columna "Dif." la existencia o no de diferencias significativas a favor del algoritmo cultural propuesto, y en las columnas "Prom. AG" y "Prom. AC" los promedios de los resultados obtenidos por instancia de los algoritmos genético y cultural; en la columna "Valor p" se muestran los valores p obtenidos por la prueba de *Wilcoxon*. Esto utilizando las instancias de tamaño 100. Las existencias de diferencias de significación de los resultados con las instancias de tamaño 50, 150 y 200, se pueden observar en el apéndice A, donde se observa que el algoritmo cultural alcanza mejores resultados que el algoritmo genético simple, puesto que la prueba de *Wilcoxon* obtiene diferencias de significación a favor del algoritmo cultural. Así también, en la Tabla 5.6 se muestra el resumen de la prueba de *Wilcoxon* con todas las instancias.

No. Nodos	Diferencia significativa a favor del AC	Diferencia significativa en contra del AC	Sin diferencia significativa entre los algoritmos comparados
10	28	0	72
50	89	0	11
100	100	0	0
150	100	0	0
200	100	0	0

Tabla 5.6: Resumen de la prueba de *Wilcoxon*.

5.7 Resumen del capítulo

En este capítulo se discutieron los resultados de tres algoritmos, a saber: el algoritmo genético, el algoritmo cultural propuesto y el algoritmo voraz. Se observó que los resultados del algoritmo cultural propuesto mejoran a los del genético simple; así también, se observó cómo los casos de éxito de cada una de las fuentes de conocimiento del algoritmo cultural propuesto, varían tanto por generación como en cada tamaño de instancia; teniendo una mayor tasa de éxito tanto la fuente de conocimiento de dominio con costos, como la fuente de conocimiento normativo. Se realizó la comparación de los resultados entre los tres algoritmos y se observó cómo el algoritmo cultural propuesto mejora al algoritmo voraz del estado del arte.

Inst.	Prom.AG	Prom.AC	Valor p	Dif.	Inst.	Prom.AG	Prom.AC	Valor p	Dif.
1	15.42831	12.73786	1.86E-08	sí	51	17.1631	14.99825	1.86E-09	sí
2	11.7125	10.13333	8.01E-08	sí	52	18.02673	15.52648	1.86E-09	sí
3	15.3887	12.72262	1.86E-09	sí	53	14.18264	12.23302	3.73E-09	sí
4	17.07326	14.77612	1.86E-09	sí	54	13.11113	11.32098	5.59E-09	sí
5	12.559	10.68831	1.86E-09	sí	55	11.39924	9.222283	1.86E-09	sí
6	13.11903	11.22293	3.73E-09	sí	56	17.44043	15.33629	1.86E-09	sí
7	13.30221	11.00587	3.54E-08	sí	57	14.26353	12.51762	9.31E-09	sí
8	14.65048	13.01429	9.98E-07	sí	58	13.28832	11.1302	1.86E-09	sí
9	15.52548	13.32731	1.86E-09	sí	59	13.43761	11.29932	3.73E-09	sí
10	12.44924	10.55697	1.86E-09	sí	60	14.28416	11.7002	1.86E-09	sí
11	13.60399	11.80014	3.73E-09	sí	61	15.29272	13.68325	1.86E-09	sí
12	13.26598	11.22613	1.86E-09	sí	62	16.02989	13.71029	1.86E-09	sí
13	14.34454	12.17798	1.86E-09	sí	63	13.97473	10.98472	1.86E-09	sí
14	15.02194	12.72467	1.86E-09	sí	64	13.94831	12.49824	3.73E-09	sí
15	15.59069	14.06969	5.59E-09	sí	65	16.23226	14.34674	1.86E-09	sí
16	14.42061	12.05031	1.86E-09	sí	66	12.61447	10.54647	1.86E-09	sí
17	13.42691	11.76091	1.86E-08	sí	67	14.77003	13.48082	2.61E-08	sí
18	15.11805	13.58801	4.66E-08	sí	68	12.15908	10.33648	1.86E-09	sí
19	15.09703	13.93286	4.97E-05	sí	69	15.66991	13.42334	1.86E-09	sí
20	11.99185	10.20631	1.86E-09	sí	70	14.59944	12.77093	1.86E-09	sí
21	11.04929	8.631025	1.86E-09	sí	71	13.79479	11.2342	3.73E-09	sí
22	12.77284	10.5285	1.86E-09	sí	72	14.27764	11.85661	5.59E-09	sí
23	14.15771	12.12824	3.54E-08	sí	73	14.95902	12.3569	1.86E-09	sí
24	12.36237	10.87244	2.61E-08	sí	74	11.62693	9.649472	9.31E-09	sí
25	10.07409	8.161465	1.86E-09	sí	75	17.43649	15.32058	9.31E-09	sí
26	14.85359	12.52592	5.59E-09	sí	76	13.01091	11.42511	1.86E-08	sí
27	13.48217	11.23347	3.73E-09	sí	77	11.73392	10.23257	5.59E-09	sí
28	12.03672	10.40042	1.86E-09	sí	78	15.22272	12.39054	3.73E-09	sí
29	12.48352	10.41917	1.86E-09	sí	79	12.72918	11.01231	8.01E-08	sí
30	13.26037	10.49798	1.86E-09	sí	80	15.28101	12.92469	1.86E-09	sí
31	16.54063	14.8529	3.54E-08	sí	81	13.03544	10.6978	1.86E-09	sí
32	11.9723	9.106468	1.86E-09	sí	82	15.49083	12.93139	1.86E-09	sí
33	13.77766	11.48989	2.61E-08	sí	83	14.83759	12.74492	1.30E-08	sí
34	18.02303	14.99706	1.86E-09	sí	84	15.66814	13.83603	6.15E-08	sí
35	16.34929	14.54184	2.61E-08	sí	85	13.19523	10.9561	3.73E-09	sí
36	14.66588	12.66896	1.86E-09	sí	86	15.8505	13.63955	3.73E-09	sí
37	13.92281	11.92818	1.86E-09	sí	87	13.82781	11.65912	1.86E-09	sí
38	16.96765	14.83395	1.86E-09	sí	88	13.45509	11.35441	9.31E-09	sí
39	18.38665	16.22562	3.73E-09	sí	89	12.4789	10.61962	9.31E-09	sí
40	13.83382	11.56959	1.86E-09	sí	90	15.5567	13.26281	1.86E-09	sí
41	13.67635	11.79568	1.86E-09	sí	91	12.31292	10.86909	1.19E-06	sí
42	16.7873	14.42417	2.61E-08	sí	92	14.20294	12.33139	9.31E-09	sí
43	14.68677	12.91479	2.55E-07	sí	93	14.95406	13.04659	1.86E-09	sí
44	17.0619	14.58499	1.86E-09	sí	94	15.60661	14.14409	2.61E-08	sí
45	14.35534	12.4808	8.01E-08	sí	95	14.02451	12.42313	2.05E-07	sí
46	11.36328	9.662391	3.54E-08	sí	96	14.45212	12.28112	2.61E-08	sí
47	14.46802	11.85263	1.86E-09	sí	97	14.41332	12.16093	1.86E-09	sí
48	13.71743	11.82441	1.86E-09	sí	98	13.9449	12.32619	2.61E-08	sí
49	11.96935	9.51656	1.86E-09	sí	99	13.86813	11.4743	1.86E-09	sí
50	15.88008	13.59011	1.86E-09	sí	100	12.43858	10.16814	1.86E-08	sí

Tabla 5.7: Prueba de *Wilcoxon* con las instancias de tamaño 100.

6

Conclusiones

6.1 Conclusiones

En esta tesis se mostró cómo el problema del EVCSP puede ser abordado bajo el enfoque del cómputo evolutivo. Inicialmente, se realizaron experimentaciones con un algoritmo genético simple, dado que no se contaba en la literatura con una metaheurística que pudiera ser empleada como punto de referencia para las comparaciones con el algoritmo cultural propuesto en este trabajo de investigación. Una de las aportaciones de la experimentación con dicha metaherística genérica, aparte de ser un punto de partida para el diseño de la propuesta, fue obtener las funciones de penalización para el manejo de las restricciones del problema.

En este trabajo de investigación se describieron los aspectos importantes y los motivos que se tomaron en cuenta para realizar el diseño de las fuentes de conocimiento empleadas en el algoritmo cultural propuesto, obteniéndose un diseño con cuatro fuentes de conocimiento, a saber: dos fuentes de conocimiento de dominio (una con respecto a los costos de construcción y la otra a los puntos

de articulación), una tercera fuente de conocimiento fue la situacional y finalmente una cuarta de conocimiento normativo.

La extracción del conocimiento durante la búsqueda realizada por el algoritmo propuesto aportó mejoras al desempeño del mismo, lo cual se corroboró con la experimentación realizada, en donde el algoritmo propuesto mejoró los promedios obtenidos por el algoritmo voraz del estado del arte a partir de los resultados con las 100 instancias por cada tamaño.

Esto nos indica que emplear el conocimiento de las tendencias del espacio de creencias ayuda en la obtención de mejores resultados, y que el algoritmo cultural es conveniente para tratar el problema del EVCSP, lo cual permitió comprobar la hipótesis formulada en este trabajo de investigación.

Sin embargo, se observó que la convergencia del algoritmo propuesto se estancaba cuando el proceso evolutivo se encontraba en las etapas finales de la búsqueda, lo cual se podría atribuir tanto a una baja capacidad de exploración de los operadores utilizados como a una alta presión selectiva.

6.2 Trabajo futuro

Como trabajo futuro, en primera instancia, hablando del diseño del algoritmo propuesto, sería interesante emplear una fuente de conocimiento histórico del algoritmo cultural, en la cual se creé una lista de eventos históricos importantes en la búsqueda para que los individuos puedan usar este conocimiento como guía para la selección de una dirección de movimiento.

También, con respecto al diseño del algoritmo, habría que modificar el espacio poblacional utilizado, para llevar un control adecuado de la presión de selección y para mejorar la velocidad de convergencia y, de este modo, intentar mejorar las soluciones obtenidas por el algoritmo.

Por lo que se refiere a la experimentación, queda pendiente probar la solución propuesta con instancias de grafos que modelen ciudades, ya que sólo se realizaron pruebas con instancias sintéticas.

Finalmente, en cuanto al modelado del problema, falta explorar la viabilidad de incorporar otras restricciones, tales como: costos de operación y flujo de tráfico.

A

Tablas de resultados

En las tablas A.1,A.2,A.3,A.4,A.5 se muestran los resultados obtenidos tanto con el algoritmo voraz como con el algoritmo cultural propuesto, utilizando las 100 instancias de tamaño 10, 50,100, 150 y 200 respectivamente.

En las tablas A.6, A.7, A.8 y A.9 se muestra la comparación entre el algoritmo genético y el algoritmo cultural propuesto, con las 100 instancias de tamaños 10, 50, 150 y 200 respectivamente, y los valores p con la prueba de *Wilcoxon*.

Inst.	AV	AC				Inst.	AV	AC			
	Mejor	Promedio	Mediana	Mejor	EC		Mejor	Promedio	Mediana	Mejor	EC
1	0.995	0.995	0.995	0.995	2	51	1.468	1.103	1.103	**1.103**	2
2	1.703	1.702	1.702	**1.702**	4	52	1.388	1.387	1.387	**1.387**	2
3	1.753	1.752	1.752	**1.752**	3	53	1.499	1.498	1.498	**1.498**	4
4	1.976	1.976	1.976	1.976	4	54	0.167	0.166	0.166	**0.166**	2
5	1.825	1.824	1.824	**1.824**	4	55	4.813	4.813	4.813	4.813	6
6	0.763	0.715	0.700	**0.700**	3	56	2.221	2.221	2.221	2.221	3
7	2.815	2.814	2.814	**2.814**	6	57	1.313	1.312	1.312	**1.312**	3
8	0.690	0.691	0.690	0.690	4	58	1.855	1.622	1.622	**1.622**	3
9	0.679	0.704	0.679	0.679	4	59	0.9	0.899	0.899	**0.899**	4
10	2.388	2.388	2.388	2.388	4	60	1.073	1.083	1.073	1.073	4
11	1.258	1.257	1.257	**1.257**	3	61	0.893	0.893	0.893	0.893	3
12	1.067	1.066	1.066	**1.066**	4	62	0.24	0.239	0.239	**0.239**	3
13	1.225	1.224	1.224	**1.224**	3	63	1.499	1.498	1.498	**1.498**	4
14	3.213	3.213	3.213	3.213	5	64	0.389	0.399	0.389	0.389	3
15	2.178	2.178	2.178	2.178	4	65	1.229	1.229	1.229	1.229	2
16	1.702	1.702	1.702	1.702	4	66	3.149	2.772	2.745	**2.745**	5
17	2.681	2.681	2.681	2.681	5	67	0.928	0.927	0.927	**0.927**	4
18	1.481	1.481	1.481	1.481	3	68	1.82	1.819	1.819	**1.819**	4
19	1.415	1.415	1.415	1.415	4	69	0.786	0.785	0.785	**0.785**	4
20	2.732	2.732	2.732	2.732	5	70	1.105	1.105	1.105	1.105	3
21	1.998	1.058	1.058	**1.058**	2	71	1.324	1.324	1.324	1.324	3
22	1.453	1.452	1.452	**1.452**	4	72	3.239	3.239	3.239	3.239	6
23	3.205	3.205	3.205	3.205	5	73	1.733	1.743	1.733	1.733	3
24	2.272	2.272	2.272	2.272	4	74	0.923	0.922	0.922	**0.922**	3
25	0.581	0.581	0.581	0.581	2	75	1.837	1.837	1.837	1.837	4
26	1.896	1.895	1.895	**1.895**	4	76	1.661	1.660	1.660	**1.660**	4
27	1.459	1.459	1.459	1.459	2	77	2.951	2.950	2.950	**2.950**	6
28	0.612	0.611	0.611	**0.611**	4	78	1.484	1.484	1.484	1.484	5
29	2.107	2.113	2.107	2.107	3	79	1.38	1.379	1.379	**1.379**	3
30	3.449	3.449	3.449	3.449	6	80	0.997	0.524	0.524	**0.524**	2
31	0.752	0.751	0.751	**0.751**	2	81	2.245	2.245	2.245	2.245	5
32	1.358	1.358	1.358	1.358	4	82	0.732	0.732	0.732	0.732	3
33	3.373	3.373	3.373	3.373	5	83	1.504	1.504	1.504	1.504	4
34	1.332	1.331	1.331	**1.331**	4	84	0.787	0.787	0.787	0.787	4
35	0.573	0.573	0.573	0.573	3	85	1.717	1.717	1.717	1.717	4
36	1.901	1.900	1.900	**1.900**	5	86	2.694	2.399	2.389	**2.389**	3
37	0.966	0.966	0.966	0.966	2	87	1.894	1.893	1.893	**1.893**	4
38	0.827	0.826	0.826	**0.826**	3	88	0.265	0.264	0.264	**0.264**	2
39	0.606	0.605	0.605	**0.605**	3	89	0.955	0.955	0.955	0.955	3
40	0.933	0.933	0.933	0.933	3	90	1.405	0.994	0.930	**0.930**	2
41	1.117	1.117	1.117	1.117	3	91	0.971	0.971	0.971	0.971	3
42	0.667	0.667	0.667	0.667	2	92	1.891	1.890	1.890	**1.890**	3
43	1.91	1.917	1.909	**1.909**	5	93	2.292	2.292	2.292	2.292	4
44	1.065	1.065	1.065	1.065	3	94	2.016	1.417	1.417	**1.417**	3
45	1.733	1.733	1.733	1.733	4	95	2.437	2.436	2.436	**2.436**	4
46	0.692	0.691	0.691	**0.691**	3	96	2.307	2.307	2.307	2.307	5
47	2.654	2.653	2.653	**2.653**	5	97	1.514	1.514	1.514	1.514	5
48	3.378	3.377	3.377	**3.377**	6	98	1.169	1.168	1.168	**1.168**	3
49	1.258	1.270	1.258	1.258	5	99	0.844	0.835	0.834	**0.834**	2
50	1.418	1.418	1.418	1.418	4	100	2.069	2.068	2.068	**2.068**	4
	Promedios de los valores de las 100 instancias:						1.600	1.564		1.562	

Tabla A.1: Comparación de los resultados entre los algoritmos voraz y cultural con las 100 instancias de tamaño 10, donde los valores en negritas en la columna "Mejor" son los que mejoran al algoritmo voraz.

Inst.	AV	AC				Inst.	AV	AC			
	Mejor	Promedio	Mediana	Mejor	EC		Mejor	Promedio	Mediana	Mejor	EC
1	7.669	7.738	7.688	**7.668**	19	51	5.59	5.690	5.590	**5.589**	20
2	7.456	7.290	7.163	**6.824**	16	52	8.516	8.544	8.383	**8.349**	18
3	9.009	9.043	9.009	**9.008**	22	53	6.258	6.171	6.028	**5.729**	16
4	5.892	6.161	6.073	**5.891**	18	54	9.552	9.686	9.552	**9.459**	20
5	6.747	6.721	6.703	**6.585**	17	55	6.634	6.857	6.667	**6.398**	20
6	6.893	6.759	6.595	**6.594**	17	56	7.237	7.356	7.237	**7.187**	21
7	6.645	6.859	6.645	6.645	17	57	8.574	8.485	8.574	**8.268**	18
8	7.578	7.220	7.079	**6.987**	19	58	6.7	6.489	6.568	**6.124**	18
9	7.528	7.528	7.528	**7.527**	20	59	4.822	4.906	4.822	**4.591**	14
10	4.843	4.929	4.843	**4.842**	18	60	7.83	8.117	7.970	**7.772**	21
11	6.001	6.030	5.921	**5.752**	17	61	5.543	5.109	4.887	**4.886**	18
12	5.181	5.660	5.407	**5.180**	18	62	7.345	7.446	7.345	**7.241**	21
13	6.171	5.580	5.400	**5.399**	15	63	9.415	9.355	9.372	**9.241**	20
14	8.819	8.819	8.819	8.819	19	64	6.489	6.513	6.489	**6.110**	18
15	10.352	10.429	10.352	**9.817**	21	65	5.869	6.206	6.024	**5.738**	18
16	6.989	6.840	6.926	**6.624**	18	66	6.902	7.104	7.007	**6.901**	21
17	7.897	8.148	8.004	7.897	22	67	11.042	10.845	10.858	**10.479**	21
18	6.504	6.570	6.583	**6.291**	15	68	8.895	8.918	8.895	**8.894**	21
19	6.888	6.753	6.739	**6.587**	20	69	8.439	8.544	8.466	**8.438**	18
20	10.927	10.872	10.927	**10.429**	20	70	7.468	7.679	7.496	**7.370**	21
21	6.381	6.381	6.381	6.381	17	71	7.084	6.978	6.886	**6.788**	17
22	7.336	7.520	7.336	**7.335**	18	72	7.967	8.033	7.967	7.967	21
23	7.047	7.584	7.232	7.047	20	73	6.784	6.698	6.677	**6.434**	16
24	5.102	5.210	5.102	**5.057**	17	74	7.276	7.228	7.254	**7.085**	19
25	6.224	6.195	6.224	**5.978**	19	75	8.509	8.125	7.882	**7.802**	18
26	6.492	6.546	6.492	**6.491**	19	76	7.717	8.030	7.938	**7.690**	18
27	10.231	10.274	10.264	**9.739**	20	77	9.769	9.257	9.239	**8.355**	18
28	9.211	9.242	9.211	**8.946**	21	78	9.003	8.788	8.772	**8.191**	20
29	6.034	6.035	6.034	**5.901**	18	79	6.427	6.182	5.922	**5.813**	17
30	8.348	8.394	8.178	**8.177**	19	80	7.965	7.831	7.795	**7.266**	20
31	7.229	7.340	7.229	**7.228**	19	81	7.258	6.919	6.816	**6.815**	18
32	7.919	7.716	7.380	**7.370**	17	82	6.374	6.524	6.460	**6.373**	19
33	7.08	6.574	6.608	**6.254**	17	83	7.329	7.217	7.240	**7.160**	19
34	6.746	6.887	6.748	6.746	19	84	7.501	7.657	7.501	7.501	18
35	5.831	5.635	5.590	**5.589**	15	85	4.78	4.905	4.780	4.780	16
36	5.733	5.985	5.911	**5.414**	17	86	7.063	7.057	7.019	**6.922**	19
37	5.075	5.338	5.017	**5.016**	17	87	7.457	8.006	7.928	**7.456**	22
38	7.418	7.588	7.515	7.418	20	88	7.878	8.122	8.002	**7.877**	18
39	8.679	7.733	7.647	**7.288**	14	89	6.255	5.973	5.820	**5.820**	18
40	9.389	9.444	9.414	**8.959**	18	90	6.314	6.560	6.314	**6.313**	19
41	7.697	7.755	7.697	**7.696**	18	91	6.389	6.389	6.389	**6.268**	18
42	6.838	6.702	6.639	**6.383**	16	92	7.369	7.451	7.369	**7.290**	17
43	9.947	9.889	9.910	**9.718**	20	93	7.101	7.101	7.101	7.101	15
44	7.592	7.592	7.592	**7.591**	18	94	7.433	7.623	7.433	7.433	21
45	6.184	6.219	6.097	**6.096**	15	95	5.492	5.527	5.492	**5.491**	18
46	6.825	6.836	6.807	**6.349**	19	96	6.566	6.657	6.566	**6.565**	18
47	7.723	7.814	7.793	7.723	18	97	10.539	10.555	10.539	**10.429**	20
48	8.569	8.678	8.651	8.569	20	98	5.733	6.127	6.133	**5.533**	19
49	7.839	7.961	7.839	**7.838**	19	99	9.009	9.141	9.009	**9.008**	20
50	7.93	8.150	7.930	**7.929**	17	100	7.208	7.330	7.208	**7.172**	18
	Promedios de los valores de las 100 instancias:						7.353	7.371		7.131	

Tabla A.2: Comparación de los resultados entre los algoritmos voraz y cultural con las 100 instancias de tamaño 50, donde los valores en negritas en la columna "Mejor" son los que mejoran al algoritmo voraz.

Inst.	AV	AC				Inst.	AV	AC			
	Mejor	Promedio	Mediana	Mejor	EC		Mejor	Promedio	Mediana	Mejor	EC
1	13.143	12.737	12.485	**11.880**	33	51	14.347	14.998	14.511	**14.346**	38
2	9.714	10.133	10.025	**9.342**	32	52	15.496	15.526	15.503	**15.146**	37
3	12.797	12.722	12.600	**12.047**	35	53	11.817	12.233	12.125	**11.639**	35
4	15.134	14.776	14.801	**14.200**	36	54	11.221	11.320	11.221	**10.612**	33
5	10.096	10.688	10.649	10.289	31	55	9.197	9.222	9.157	**8.658**	33
6	10.853	11.222	11.197	**10.661**	33	56	14.725	15.336	15.235	**14.556**	38
7	10.694	11.005	11.018	**10.219**	32	57	12.922	12.517	12.420	**11.865**	34
8	13.093	13.014	12.966	**12.152**	34	58	10.94	11.130	11.029	**10.741**	32
9	12.706	13.327	13.206	**12.705**	36	59	11.123	11.299	11.277	**10.769**	34
10	10.814	10.556	10.453	**9.843**	33	60	11.992	11.700	11.708	**11.235**	34
11	11.738	11.800	11.738	**11.447**	31	61	13.993	13.683	13.584	**13.141**	34
12	10.964	11.226	11.204	**10.539**	34	62	13.725	13.710	13.634	**13.357**	35
13	11.387	12.177	11.995	**11.197**	35	63	10.684	10.984	10.772	**10.554**	34
14	13.105	12.724	12.714	**12.242**	34	64	12.565	12.498	12.457	**11.895**	30
15	14.461	14.069	13.967	**13.430**	33	65	13.846	14.346	14.444	**13.329**	35
16	11.772	12.050	11.987	**11.383**	35	66	10.116	10.546	10.499	**10.115**	33
17	12.188	11.760	11.748	**11.086**	31	67	13.265	13.480	13.232	**12.889**	36
18	13.601	13.588	13.387	**12.859**	32	68	10.666	10.336	10.189	**9.925**	30
19	14.379	13.932	13.791	**12.956**	35	69	13.059	13.423	13.284	**12.680**	32
20	9.999	10.206	10.154	**9.929**	30	70	12.443	12.770	12.710	**12.356**	31
21	8.093	8.631	8.296	8.101	32	71	10.912	11.234	10.924	**10.682**	37
22	9.890	10.528	10.548	**9.406**	35	72	11.612	11.856	11.777	**11.387**	34
23	12.441	12.128	12.089	**11.464**	33	73	12.373	12.356	12.373	**11.971**	32
24	11.008	10.872	10.906	**10.254**	34	74	9.352	9.649	9.476	**9.080**	31
25	7.792	8.161	8.030	**7.558**	30	75	15.551	15.320	15.098	**14.271**	35
26	12.783	12.525	12.534	**11.987**	33	76	11.03	11.425	11.332	**10.869**	31
27	10.955	11.233	10.956	**10.610**	32	77	10.293	10.232	10.227	**9.374**	34
28	10.432	10.400	10.442	**9.595**	33	78	11.569	12.390	12.056	**11.511**	34
29	11.15	10.419	10.442	**9.785**	32	79	10.557	11.012	11.035	**9.988**	34
30	10.631	10.497	10.424	**9.914**	33	80	12.502	12.924	12.797	**12.501**	38
31	14.99	14.852	14.827	**13.977**	33	81	10.859	10.697	10.683	**10.505**	34
32	9.034	9.106	9.023	**8.862**	32	82	12.794	12.931	12.813	**12.316**	36
33	11.107	11.489	11.360	**10.971**	33	83	12.426	12.744	12.625	**12.034**	37
34	14.782	14.997	14.940	**14.335**	32	84	13.553	13.836	13.784	**13.049**	33
35	14.593	14.541	14.528	**13.628**	36	85	10.757	10.956	10.715	**9.956**	32
36	12.256	12.668	12.568	12.256	37	86	13.835	13.639	13.575	**13.031**	35
37	12.047	11.928	11.888	**11.728**	32	87	11.894	11.659	11.613	**10.939**	33
38	14.752	14.833	14.752	14.752	36	88	11.101	11.354	11.185	**10.598**	34
39	15.66	16.225	16.135	15.660	35	89	10.774	10.619	10.566	**10.158**	29
40	10.992	11.569	11.438	**10.567**	36	90	13.129	13.262	13.114	**12.874**	36
41	11.281	11.795	11.663	**11.144**	32	91	10.696	10.869	10.628	**10.255**	32
42	15.094	14.424	14.225	**13.930**	32	92	11.842	12.331	12.165	11.842	30
43	12.664	12.914	12.663	**12.096**	32	93	12.927	13.046	12.927	**12.906**	35
44	15.067	14.584	14.495	**13.866**	35	94	14.397	14.144	14.021	**13.237**	33
45	12.007	12.480	12.423	**11.911**	32	95	12.812	12.423	12.238	**11.793**	32
46	9.431	9.662	9.431	**9.364**	32	96	11.945	12.281	12.048	**11.298**	36
47	11.569	11.852	11.700	**11.372**	35	97	12.059	12.160	12.086	**11.545**	34
48	11.761	11.824	11.740	**10.906**	35	98	13.48	12.326	12.188	**11.578**	31
49	9.763	9.516	9.419	**9.419**	29	99	11.192	11.474	11.404	**11.087**	31
50	13.374	13.590	13.432	**12.739**	33	100	10.007	10.168	10.032	**9.506**	33
	Promedios de los valores de las 100 instancias:						12.064	12.163		11.545	

Tabla A.3: Comparación de los resultados entre los algoritmos voraz y cultural con las 100 instancias de tamaño 100, donde los valores en negritas en la columna "Mejor" son los que mejoran al algoritmo voraz.

Inst.	AV	AC				Inst.	AV	AC			
	Mejor	Promedio	Mediana	Mejor	EC		Mejor	Promedio	Mediana	Mejor	EC
1	12.307	12.320	12.305	**11.521**	35	51	5.796	6.245	6.083	**5.568**	31
2	8.262	8.632	8.611	**7.674**	37	52	7.775	7.566	7.454	**6.954**	36
3	12.133	11.591	11.452	**10.88**	37	53	8.532	9.007	8.867	**8.310**	38
4	11.500	11.255	11.264	**9.327**	34	54	9.226	9.506	9.421	**8.428**	34
5	8.330	8.909	8.836	**8.193**	36	55	9.888	10.513	10.406	9.888	36
6	10.078	9.923	9.801	**9.320**	35	56	9.041	9.022	8.912	**8.453**	37
7	9.997	10.239	10.032	**9.257**	35	57	8.860	9.985	9.880	**8.746**	37
8	8.851	9.528	9.478	**8.636**	39	58	9.784	10.155	10.110	**9.396**	34
9	9.598	10.505	10.33	9.713	36	59	11.154	11.783	11.676	**10.462**	37
10	7.327	8.159	7.964	**6.783**	39	60	7.054	7.286	7.280	**6.763**	36
11	6.596	7.165	7.161	**6.400**	35	61	10.831	10.823	10.747	**9.841**	33
12	8.751	8.875	8.815	**7.999**	33	62	9.322	9.406	9.495	**8.548**	35
13	7.695	8.011	7.782	**7.235**	37	63	8.981	9.570	9.388	**8.768**	35
14	8.367	8.521	8.284	**7.651**	36	64	8.023	8.803	8.569	**7.617**	36
15	9.815	10.978	10.953	10.099	36	65	11.133	11.704	11.639	**10.588**	37
16	8.529	9.569	9.421	8.730	37	66	10.705	11.302	11.184	**9.730**	40
17	7.906	8.255	8.208	**7.789**	35	67	10.059	10.493	10.415	**9.665**	32
18	10.757	10.801	10.716	**10.326**	35	68	8.308	9.421	9.259	8.660	34
19	8.996	9.715	9.401	**8.665**	33	69	9.292	9.762	9.718	**8.749**	38
20	8.578	9.074	9.031	**8.269**	39	70	8.942	8.891	8.669	**8.214**	36
21	7.309	7.830	7.798	**7.111**	35	71	10.943	11.094	11.062	**10.516**	37
22	9.458	9.842	9.723	**8.728**	36	72	10.185	10.666	10.572	**9.841**	35
23	8.151	8.379	8.382	**7.810**	36	73	9.117	9.586	9.568	**9.009**	33
24	7.603	8.132	8.072	**7.507**	39	74	7.773	8.078	7.895	**7.300**	33
25	9.684	10.606	10.65	**9.565**	36	75	7.563	7.793	7.608	**7.181**	37
26	8.102	9.090	8.962	8.369	37	76	9.78	10.347	10.129	**9.499**	38
27	10.920	11.05	10.936	**10.177**	42	77	11.098	11.157	11.011	**10.145**	39
28	11.019	11.633	11.655	11.028	34	78	8.221	8.825	8.514	**7.696**	36
29	8.126	8.855	8.765	**8.125**	39	79	9.949	9.197	9.104	**8.312**	35
30	8.795	9.549	9.367	**8.293**	37	80	8.904	8.708	8.577	**7.884**	30
31	9.171	9.557	9.447	**8.915**	35	81	8.251	8.816	8.637	**8.219**	37
32	7.744	8.840	8.600	**7.722**	38	82	9.619	9.767	9.546	**9.055**	35
33	9.812	9.772	9.642	**9.031**	35	83	8.791	9.599	9.431	**8.593**	40
34	8.792	9.505	9.423	**8.739**	37	84	9.264	9.570	9.581	**9.228**	33
35	8.792	9.114	9.099	**8.242**	36	85	8.561	9.134	9.010	**8.551**	38
36	8.800	10.074	9.979	9.288	39	86	9.233	9.187	9.101	**8.408**	34
37	9.762	9.800	9.805	**9.034**	36	87	13.054	13.129	13.044	**12.435**	40
38	8.797	9.079	8.952	**8.531**	33	88	11.375	11.802	11.577	**10.869**	36
39	11.107	11.654	11.54	**10.802**	36	89	10.011	10.904	10.818	**9.760**	39
40	9.363	9.899	9.873	**9.280**	36	90	9.818	10.334	10.394	**9.568**	37
41	9.552	9.592	9.471	**8.873**	33	91	12.093	12.171	12.119	**11.630**	38
42	8.818	9.044	9.162	**8.304**	34	92	9.181	9.562	9.450	**8.738**	35
43	9.867	9.643	9.525	**8.856**	35	93	11.104	11.218	11.157	**10.298**	37
44	8.969	9.020	8.843	**8.263**	36	94	7.825	8.374	8.254	**7.479**	33
45	9.636	10.065	9.972	**9.382**	39	95	11.402	11.464	11.399	**10.587**	37
46	9.259	9.819	9.606	**8.709**	36	96	11.108	11.512	11.324	**10.527**	35
47	10.577	10.659	10.647	**10.114**	34	97	7.698	8.466	8.298	**7.621**	39
48	9.229	9.652	9.745	**8.652**	32	98	8.649	9.431	9.338	**8.366**	36
49	12.626	12.592	12.441	**11.542**	35	99	7.218	7.909	7.894	**7.175**	31
50	10.611	10.919	10.689	**10.070**	37	100	11.955	12.635	12.667	**11.893**	41
	Promedios de los valores de las 100 instancias:						9.372	9.770		8.933	

Tabla A.4: Comparación de los resultados entre los algoritmos voraz y cultural con las 100 instancias de tamaño 150, donde los valores en negritas en la columna "Mejor" son los que mejoran al algoritmo voraz.

Inst.	AV	AC				Inst.	AV	AC			
	Mejor	Promedio	Mediana	Mejor	EC		Mejor	Promedio	Mediana	Mejor	EC
1	6.748	7.138	6.980	**5.955**	36	51	6.271	6.658	6.429	**5.954**	36
2	7.475	7.775	7.538	**7.227**	35	52	8.277	9.050	8.863	**7.933**	39
3	7.177	7.400	7.154	**6.697**	33	53	7.770	8.866	8.732	7.848	38
4	5.457	6.311	6.241	5.486	37	54	8.716	9.136	9.040	**8.475**	37
5	7.234	7.506	7.299	**6.735**	35	55	7.317	8.225	8.120	7.341	37
6	8.049	8.502	8.408	**7.667**	37	56	8.63	9.972	9.991	8.932	41
7	7.580	8.666	8.588	7.869	41	57	6.246	7.030	7.035	**6.129**	36
8	6.205	6.561	6.475	**5.909**	31	58	8.436	8.557	8.380	**7.792**	37
9	6.244	6.948	6.866	6.244	35	59	7.298	8.003	7.876	**7.226**	37
10	7.175	8.066	7.967	7.358	39	60	6.331	6.491	6.359	**5.923**	36
11	7.981	9.093	9.041	8.246	35	61	6.676	7.483	7.332	6.798	36
12	6.461	7.301	7.267	**6.373**	35	62	7.059	7.308	7.150	**6.218**	33
13	6.536	6.368	6.300	**5.615**	31	63	7.599	8.152	7.918	**7.439**	39
14	7.174	8.263	8.130	7.195	38	64	7.733	8.242	8.157	**7.604**	38
15	7.926	8.557	8.340	**7.685**	40	65	7.076	7.840	7.760	7.151	36
16	7.340	7.629	7.601	**7.081**	38	66	8.177	8.224	8.077	**7.528**	35
17	6.287	7.284	7.207	6.312	36	67	5.801	6.304	6.289	**5.512**	40
18	8.772	9.166	9.198	**8.294**	37	68	8.170	8.648	8.541	**7.830**	35
19	8.454	9.239	9.144	**8.302**	38	69	6.406	6.481	6.457	**5.444**	36
20	7.448	7.755	7.773	**6.886**	33	70	7.759	8.682	8.525	**7.686**	38
21	6.355	6.834	6.731	**6.194**	37	71	9.761	9.424	9.414	**8.573**	39
22	10.661	11.009	10.859	**9.673**	39	72	5.671	6.08	6.107	**5.372**	35
23	8.008	8.749	8.690	**7.911**	36	73	6.622	7.371	7.191	**6.481**	36
24	8.687	9.090	8.893	**8.154**	35	74	5.898	6.578	6.575	6.018	34
25	8.297	8.923	8.857	**8.021**	36	75	7.047	6.934	6.808	**5.997**	37
26	9.282	10.275	10.244	**9.051**	40	76	7.022	7.671	7.495	**6.639**	35
27	8.213	8.588	8.484	**7.649**	36	77	7.339	8.462	8.337	7.533	37
28	8.316	8.738	8.779	**7.935**	38	78	8.175	8.722	8.491	**7.962**	37
29	7.029	7.098	7.018	**6.777**	38	79	6.592	7.220	7.152	**6.500**	36
30	9.53	9.708	9.574	**8.886**	36	80	7.688	8.217	8.002	**7.556**	36
31	8.11	8.735	8.648	**7.716**	37	81	9.151	9.988	9.817	**8.880**	37
32	6.186	7.118	6.997	6.303	38	82	7.686	8.204	8.161	**7.607**	39
33	5.146	5.938	5.937	5.160	34	83	6.993	7.528	7.387	**6.544**	35
34	6.539	7.479	7.568	**6.286**	38	84	6.736	7.436	7.407	**6.333**	37
35	6.717	7.843	7.857	**6.627**	37	85	6.330	6.877	6.695	**6.279**	35
36	6.166	6.856	6.713	6.201	39	86	7.747	8.339	8.270	**7.681**	35
37	8.223	8.415	8.396	**7.603**	39	87	7.238	7.808	7.504	**6.605**	40
38	7.812	8.650	8.506	**7.424**	38	88	6.234	7.002	6.830	**6.193**	38
39	6.546	7.445	7.373	6.799	39	89	7.02	7.848	7.769	7.032	39
40	7.867	8.177	8.134	**7.396**	35	90	8.338	9.066	8.966	**8.064**	36
41	8.552	9.057	8.915	**8.192**	36	91	6.967	7.707	7.596	**6.859**	38
42	6.131	7.183	6.992	6.493	37	92	6.142	7.034	6.988	**6.132**	36
43	9.058	9.342	9.325	**8.151**	38	93	7.769	8.527	8.412	7.896	36
44	7.036	7.687	7.472	7.043	39	94	8.299	8.216	8.120	**7.626**	37
45	7.245	7.448	7.310	**7.017**	34	95	6.661	7.155	6.943	**6.545**	39
46	7.586	8.237	8.171	**7.557**	40	96	5.022	5.924	5.817	5.157	36
47	5.857	6.143	6.018	**5.543**	35	97	8.814	9.417	9.427	**8.601**	39
48	5.652	6.647	6.635	5.726	38	98	6.222	7.259	7.276	**6.201**	35
49	7.947	8.009	7.830	**6.791**	36	99	5.632	6.186	6.136	5.664	36
50	7.012	7.390	7.224	**6.414**	36	100	7.582	8.173	8.131	**7.387**	37
	Promedios de los valores de las 100 instancias:						7.316	7.901		7.065	

Tabla A.5: Comparación de los resultados entre los algoritmos voraz y cultural con las 100 instancias de tamaño 200, donde los valores en negritas en la columna "Mejor" son los que mejoran al algoritmo voraz.

Inst.	Prom.AG	Prom.AC	Valor p	Dif.	Inst.	Prom.AG	Prom.AC	Valor p	Dif.
1	1.02155	0.995484	1	no	51	1.176564	1.10379	0.019656157	sí
2	1.702872	1.702872	-	no	52	1.40525	1.387949	1	no
3	1.752537	1.752537	-	no	53	1.522701	1.498723	0.345778586	no
4	1.979646	1.976492	1	no	54	0.1751891	0.16668	0.173568167	no
5	1.824887	1.824887	-	no	55	4.836667	4.813494	0.173568167	no
6	0.7423397	0.7150392	0.000362568	sí	56	2.221187	2.221187	-	no
7	2.891083	2.81483	0.008610127	sí	57	1.342213	1.312521	1	no
8	0.7379318	0.6911017	6.37E-05	sí	58	1.710636	1.622015	0.002741022	sí
9	0.823117	0.7041513	0.00011708	sí	59	0.9346036	0.899876	1	no
10	2.388021	2.388021	-	no	60	1.195033	1.083538	0.001089347	sí
11	1.281622	1.257606	1	no	61	0.9330464	0.8933818	0.021303176	sí
12	1.095954	1.066919	0.345778586	no	62	0.239858	0.239858	-	no
13	1.224525	1.224525	-	no	63	1.594667	1.498535	0.097512538	no
14	3.213079	3.213079	-	no	64	0.440675	0.3993782	0.0029723	sí
15	2.178806	2.178228	1	no	65	1.229315	1.229315	-	no
16	1.702436	1.702436	-	no	66	2.826439	2.772646	0.129430674	no
17	2.681032	2.681032	-	no	67	1.179	0.927858	0.001977158	sí
18	1.492891	1.481286	0.071860638	no	68	1.819946	1.819946	-	no
19	1.415132	1.415132	-	no	69	0.785505	0.785505	-	no
20	2.732209	2.732209	-	no	70	1.118996	1.105124	0.345778586	no
21	1.089393	1.058055	1	no	71	1.350304	1.324095	1	no
22	1.505403	1.452503	0.071860638	no	72	3.239283	3.239283	-	no
23	3.239854	3.205462	0.088973012	no	73	1.809262	1.743234	0.031250017	sí
24	2.342718	2.272029	0.097512538	no	74	1.043487	0.922568	0.088973012	no
25	0.581269	0.581269	-	no	75	1.905252	1.837434	0.071860638	no
26	1.953632	1.895565	0.011866217	sí	76	1.685244	1.66086	0.345778586	no
27	1.459353	1.459353	-	no	77	2.953437	2.950532	0.036888426	sí
28	0.6269731	0.611805	0.148914673	no	78	1.484198	1.484198	-	no
29	2.181404	2.113314	0.024752021	sí	79	1.415699	1.379553	0.37109337	no
30	3.44933	3.44933	-	no	80	0.5441777	0.524839	0.005962079	sí
31	0.773741	0.751633	1	no	81	2.245254	2.245254	-	no
32	1.358039	1.358039	-	no	82	0.7562011	0.732379	1	no
33	3.392376	3.373163	0.056759446	no	83	1.515804	1.504194	0.036888426	sí
34	1.331759	1.331759	-	no	84	0.8589472	0.787241	0.148914673	no
35	0.6557539	0.573334	0.097512538	no	85	1.7182	1.71735	0.109598583	no
36	2.231942	1.900592	0.000179675	sí	86	2.438868	2.399218	0.041632847	sí
37	1.024405	0.966195	0.036888426	sí	87	1.902408	1.893772	1	no
38	0.8364236	0.826752	0.019656157	sí	88	0.26826	0.264912	1	no
39	0.6103664	0.605967	1	no	89	0.9803828	0.955089	0.37109337	no
40	1.054654	0.933424	0.021303176	sí	90	1.025812	0.994192	0.565318637	no
41	1.119644	1.117329	0.148914673	no	91	1.020142	0.971066	0.03054485	sí
42	0.70845	0.667322	0.002073025	sí	92	1.916037	1.890784	1	no
43	1.925357	1.917494	0.772829993	no	93	2.30073	2.292353	1	no
44	1.13963	1.065384	0.026268291	sí	94	1.549062	1.41792	0.026268291	sí
45	1.820748	1.733398	0.013560119	sí	95	2.448808	2.436549	0.345778586	no
46	0.7228878	0.691884	1	no	96	2.317938	2.307108	0.071860638	no
47	2.653991	2.653991	-	no	97	1.51417	1.51417	-	no
48	3.381426	3.377906	0.148914673	no	98	1.18923	1.168981	0.173568167	no
49	1.407661	1.270771	0.001856719	sí	99	0.8668554	0.8353756	0.045946928	sí
50	1.424263	1.418204	0.148914673	no	100	2.068659	2.068659	-	no

Tabla A.6: Prueba de *Wilcoxon* con las instancias de tamaño 10, los "-" en la columna "Valor p" significa que las muestras son idénticas y por tanto no se puede obtener un valor p

Inst.	Prom.AG	Prom.AC	Valor p	Dif.	Inst.	Prom.AG	Prom.AC	Valor p	Dif.
1	8.11259	7.737796	1.25E-05	sí	51	6.366349	5.689959	5.89E-06	sí
2	7.57052	7.289598	0.019660816	sí	52	8.609618	8.544258	0.211522429	no
3	9.422518	9.043284	5.20E-05	sí	53	6.58241	6.171161	0.000419403	sí
4	6.49002	6.160842	0.001256768	sí	54	9.846064	9.686449	0.020110288	sí
5	6.966645	6.720895	0.000585835	sí	55	7.159405	6.857249	0.042651705	sí
6	7.124184	6.75911	0.000118526	sí	56	7.88878	7.35591	4.37E-05	sí
7	7.367532	6.858937	0.0001353	sí	57	8.807061	8.485086	0.001536986	sí
8	7.817544	7.219943	6.43E-05	sí	58	6.745338	6.489124	0.017453669	sí
9	7.909291	7.527938	8.72E-06	sí	59	5.30607	4.906181	0.000554826	sí
10	5.443865	4.929169	4.49E-06	sí	60	8.458303	8.116658	0.000592546	sí
11	6.199881	6.02991	0.059942202	no	61	5.474073	5.108951	0.001714358	sí
12	6.334008	5.660286	0.000927673	sí	62	8.348821	7.445979	6.87E-06	sí
13	5.703253	5.579679	0.115126288	no	63	9.776675	9.35496	1.82E-06	sí
14	9.136548	8.819036	5.93E-06	sí	64	7.041143	6.512747	0.000498731	sí
15	10.61269	10.42866	0.146081528	no	65	6.495844	6.206159	0.0101373	sí
16	7.141321	6.839921	0.003005309	sí	66	7.492376	7.103859	7.99E-06	sí
17	8.405061	8.148085	0.001073756	sí	67	11.21377	10.84454	0.002845051	sí
18	6.808988	6.569815	0.014545846	sí	68	9.217935	8.918335	1.24E-05	sí
19	7.1714	6.753263	5.02E-05	sí	69	8.865003	8.544455	0.001230509	sí
20	11.18144	10.872	0.002856212	sí	70	8.06274	7.679176	0.000359952	sí
21	6.569455	6.381327	1.30E-05	sí	71	7.298824	6.978053	0.001013592	sí
22	7.987495	7.519686	0.001240873	sí	72	8.201901	8.033064	5.62E-05	sí
23	8.475617	7.58374	1.82E-05	sí	73	7.003848	6.698354	0.002030546	sí
24	5.474001	5.209542	0.010537759	sí	74	7.476196	7.227656	0.000226651	sí
25	6.428543	6.194643	0.006831261	sí	75	8.312602	8.124733	0.034191076	sí
26	6.809805	6.546125	0.000489301	sí	76	8.283283	8.030038	0.004755564	sí
27	10.39345	10.27383	0.232871242	no	77	9.52091	9.25721	0.180025134	no
28	9.550607	9.242322	0.000423987	sí	78	9.037317	8.787671	0.040489722	sí
29	6.372162	6.034533	1.80E-05	sí	79	6.530966	6.182252	0.003790174	sí
30	8.972211	8.393727	0.000271601	sí	80	8.192459	7.831192	0.014377246	sí
31	7.496298	7.340034	0.004860233	sí	81	7.214612	6.919258	0.000644199	sí
32	7.928726	7.71626	0.087781669	no	82	6.92367	6.524345	6.02E-05	sí
33	6.789509	6.573592	0.097754451	no	83	7.385004	7.217265	0.00059977	sí
34	7.211641	6.887472	0.001779652	sí	84	7.885155	7.657302	0.006861444	sí
35	5.863738	5.635147	0.000188159	sí	85	5.424816	4.905326	8.53E-05	sí
36	6.093092	5.985475	0.34921138	no	86	7.299423	7.056907	0.001457477	sí
37	5.851379	5.33753	0.000236971	sí	87	8.451453	8.006444	0.000505488	sí
38	8.205295	7.588125	2.06E-05	sí	88	8.634455	8.122326	0.000174159	sí
39	7.992746	7.732585	0.130031403	no	89	6.457697	5.973159	3.98E-05	sí
40	9.550138	9.444093	0.038717878	sí	90	7.235791	6.560278	0.000473127	sí
41	8.173248	7.75453	5.01E-05	sí	91	6.597424	6.38861	0.000564837	sí
42	6.945962	6.702345	0.003222989	sí	92	7.955629	7.450996	0.000307059	sí
43	10.03493	9.889352	0.006046474	sí	93	7.310042	7.101297	4.16E-05	sí
44	8.230934	7.591748	2.87E-05	sí	94	7.84988	7.622582	0.012923352	sí
45	6.397359	6.218837	0.020884301	sí	95	5.898258	5.526664	2.13E-05	sí
46	7.131693	6.835885	0.007612137	sí	96	7.256669	6.657016	7.17E-05	sí
47	8.001919	7.813963	0.0025437	sí	97	10.79142	10.55478	0.000365625	sí
48	9.222789	8.678426	1.43E-05	sí	98	6.313732	6.126914	0.145999491	no
49	8.398696	7.961004	0.00023929	sí	99	9.472476	9.140924	0.00021357	sí
50	8.559465	8.149558	3.06E-05	sí	100	7.774296	7.330379	5.26E-05	sí

Tabla A.7: Prueba de *Wilcoxon* con las instancias de tamaño 50

Inst.	Prom.AG	Prom.AC	Valor p	Dif.	Inst.	Prom.AG	Prom.AC	Valor p	Dif.
1	15.90737	12.3201	1.86E-09	sí	51	8.9643	6.245801	1.86E-09	sí
2	11.67483	8.632887	1.86E-09	sí	52	11.44131	7.566585	1.86E-09	sí
3	14.95881	11.59181	1.86E-09	sí	53	12.55941	9.007651	1.86E-09	sí
4	15.14521	11.2555	1.86E-09	sí	54	12.78851	9.506389	1.86E-09	sí
5	12.60528	8.909471	1.86E-09	sí	55	14.59187	10.51316	1.86E-09	sí
6	14.04267	9.923333	1.86E-09	sí	56	12.99072	9.022715	1.86E-09	sí
7	13.62875	10.23971	1.86E-09	sí	57	13.61476	9.985855	1.86E-09	sí
8	13.05397	9.528974	1.86E-09	sí	58	14.04077	10.15557	1.86E-09	sí
9	14.44077	10.50572	1.86E-09	sí	59	15.13721	11.78311	1.86E-09	sí
10	11.26461	8.159857	1.86E-08	sí	60	10.80947	7.286827	1.86E-09	sí
11	10.93935	7.165668	1.86E-09	sí	61	14.19821	10.82316	1.86E-09	sí
12	13.76809	8.875828	1.86E-09	sí	62	12.44386	9.406203	1.86E-09	sí
13	11.68367	8.011587	1.86E-09	sí	63	12.99865	9.570978	1.86E-09	sí
14	12.46891	8.521992	1.86E-09	sí	64	12.66533	8.803339	1.86E-09	sí
15	14.65086	10.97893	1.86E-09	sí	65	15.22548	11.70472	1.86E-09	sí
16	12.60859	9.569977	3.73E-09	sí	66	15.02287	11.3024	1.86E-09	sí
17	11.86453	8.255112	1.86E-09	sí	67	13.62066	10.49334	1.86E-09	sí
18	14.61489	10.80105	1.86E-09	sí	68	13.25184	9.421707	1.86E-09	sí
19	13.72131	9.71567	1.86E-09	sí	69	13.84695	9.762087	1.86E-09	sí
20	12.02989	9.074395	1.86E-09	sí	70	12.18922	8.891995	1.86E-09	sí
21	11.47689	7.830525	1.86E-09	sí	71	14.64664	11.09453	1.86E-09	sí
22	13.64237	9.842173	1.86E-09	sí	72	14.14175	10.66602	1.86E-09	sí
23	11.72673	8.379281	1.86E-09	sí	73	12.87864	9.586093	1.86E-09	sí
24	12.11583	8.132887	1.86E-09	sí	74	11.08253	8.078656	1.86E-09	sí
25	13.71995	10.60619	1.86E-09	sí	75	11.45198	7.79348	1.86E-09	sí
26	12.36851	9.090411	1.86E-09	sí	76	14.28306	10.3479	1.86E-09	sí
27	14.45207	11.05003	1.86E-09	sí	77	14.74194	11.15741	5.59E-09	sí
28	15.43083	11.63374	1.86E-09	sí	78	12.40201	8.8257	1.86E-09	sí
29	14.5514	8.855998	1.86E-09	sí	79	12.86277	9.197806	1.86E-09	sí
30	13.47297	9.549998	1.86E-09	sí	80	12.23667	8.708391	1.86E-09	sí
31	13.43476	9.557566	1.86E-09	sí	81	12.01748	8.816882	1.86E-09	sí
32	12.54586	8.840352	1.86E-09	sí	82	13.36122	9.767869	1.86E-09	sí
33	13.45144	9.77227	1.86E-09	sí	83	13.74902	9.599648	1.86E-09	sí
34	12.73049	9.505059	1.86E-09	sí	84	15.27762	9.570985	1.86E-09	sí
35	12.70575	9.114734	1.86E-09	sí	85	13.15963	9.134604	1.86E-09	sí
36	13.61523	10.07472	1.86E-09	sí	86	13.20621	9.187356	1.86E-09	sí
37	13.18303	9.800131	1.86E-09	sí	87	16.36172	13.1297	1.86E-09	sí
38	12.20676	9.079852	1.86E-09	sí	88	15.45011	11.80205	1.86E-09	sí
39	14.49632	11.65419	3.73E-09	sí	89	14.36192	10.90424	1.86E-09	sí
40	13.408	9.899224	1.86E-09	sí	90	13.96509	10.33497	1.86E-09	sí
41	12.70151	9.592731	3.73E-09	sí	91	15.27516	12.17171	1.86E-09	sí
42	12.96962	9.044749	1.86E-09	sí	92	12.76103	9.562964	1.86E-09	sí
43	12.88803	9.643342	1.86E-09	sí	93	15.36344	11.2187	1.86E-09	sí
44	12.27374	9.020284	3.73E-09	sí	94	12.13111	8.374888	3.73E-09	sí
45	13.64182	10.06591	1.86E-09	sí	95	14.66011	11.46413	1.86E-09	sí
46	12.87805	9.819792	1.86E-09	sí	96	14.86496	11.51251	3.73E-09	sí
47	14.50933	10.65969	1.86E-09	sí	97	12.33046	8.466276	1.86E-09	sí
48	13.23098	9.652348	1.86E-09	sí	98	12.6091	9.431831	1.86E-09	sí
49	15.89882	12.59213	1.86E-09	sí	99	11.78848	7.909963	1.86E-09	sí
50	15.03546	10.91917	1.86E-09	sí	100	16.24414	12.63583	1.86E-09	sí

Tabla A.8: Prueba de *Wilcoxon* con las instancias de tamaño 150

Inst.	Prom.AG	Prom.AC	Valor p	Dif.	Inst.	Prom.AG	Prom.AC	Valor p	Dif.
1	11.03647	7.138921	1.86E-09	sí	51	11.39293	6.658058	1.86E-09	sí
2	13.04279	7.775798	1.86E-09	sí	52	13.89955	9.050684	1.86E-09	sí
3	12.41725	7.400329	1.86E-09	sí	53	15.01715	8.866631	1.86E-09	sí
4	10.69045	6.311741	1.86E-09	sí	54	13.2888	9.136821	1.86E-09	sí
5	12.31053	7.506643	1.86E-09	sí	55	13.14164	8.22506	1.86E-09	sí
6	13.45763	8.502282	1.86E-09	sí	56	14.11428	9.972644	3.73E-09	sí
7	12.88786	8.666928	1.86E-09	sí	57	11.50361	7.030321	1.86E-09	sí
8	10.80423	6.561253	1.86E-09	sí	58	12.66762	8.55749	1.86E-09	sí
9	11.64292	6.948753	1.86E-09	sí	59	12.74257	8.003985	1.86E-09	sí
10	12.55154	8.066045	1.86E-09	sí	60	11.19556	6.491387	1.86E-09	sí
11	13.66067	9.093254	1.86E-09	sí	61	11.82575	7.48389	1.86E-09	sí
12	12.52868	7.301521	1.86E-09	sí	62	11.47153	7.308641	1.86E-09	sí
13	10.78561	6.368112	1.86E-09	sí	63	12.39301	8.152734	1.86E-09	sí
14	13.14796	8.263448	1.86E-09	sí	64	12.81273	8.242142	1.86E-09	sí
15	12.82803	8.557595	1.86E-09	sí	65	11.9448	7.840705	1.86E-09	sí
16	11.7822	7.629839	1.86E-09	sí	66	12.86416	8.224908	1.86E-09	sí
17	11.81279	7.284484	1.86E-09	sí	67	10.67373	6.304815	1.86E-09	sí
18	13.92288	9.166321	1.86E-09	sí	68	13.32784	8.648702	1.86E-09	sí
19	13.99477	9.239445	1.86E-09	sí	69	10.51687	6.481629	1.86E-09	sí
20	12.13746	7.755024	1.86E-09	sí	70	13.60909	8.682876	1.86E-09	sí
21	12.24146	6.834636	1.86E-09	sí	71	14.33353	9.424302	1.86E-09	sí
22	15.21808	11.00939	1.86E-09	sí	72	11.01551	6.080744	1.86E-09	sí
23	13.43482	8.749804	1.86E-09	sí	73	12.63489	7.371889	1.86E-09	sí
24	14.11864	9.09068	1.86E-09	sí	74	11.38328	6.578797	1.86E-09	sí
25	14.19519	8.923029	1.86E-09	sí	75	12.09901	6.934912	1.86E-09	sí
26	15.23351	10.27546	1.86E-09	sí	76	11.63016	7.671479	1.86E-09	sí
27	13.43509	8.588771	1.86E-09	sí	77	12.18912	8.462197	1.86E-09	sí
28	13.03705	8.73827	1.86E-09	sí	78	13.3114	8.722374	1.86E-09	sí
29	11.23902	7.098897	1.86E-09	sí	79	11.77462	7.220573	1.86E-09	sí
30	14.84222	9.708403	1.86E-09	sí	80	13.46761	8.217387	1.86E-09	sí
31	13.94205	8.735858	1.86E-09	sí	81	16.0906	9.988258	1.86E-09	sí
32	11.28256	7.118095	1.86E-09	sí	82	12.78074	8.204698	1.86E-09	sí
33	10.40989	5.938533	1.86E-09	sí	83	11.67546	7.528207	1.86E-09	sí
34	12.40515	7.479783	1.86E-09	sí	84	11.72939	7.436676	1.86E-09	sí
35	12.02879	7.843025	1.86E-09	sí	85	10.76612	6.877153	1.86E-09	sí
36	11.86136	6.856579	1.86E-09	sí	86	12.04129	8.339273	1.86E-09	sí
37	12.65176	8.415911	1.86E-09	sí	87	12.41356	7.808888	1.86E-09	sí
38	12.95521	8.650022	1.86E-09	sí	88	12.28746	7.002072	1.86E-09	sí
39	12.09519	7.445422	1.86E-09	sí	89	11.97282	7.848855	1.86E-09	sí
40	12.42394	8.177058	1.86E-09	sí	90	13.68943	9.066703	1.86E-09	sí
41	14.28401	9.057906	1.86E-09	sí	91	12.38377	7.707508	1.86E-09	sí
42	11.35957	7.183445	1.86E-09	sí	92	11.35463	7.034542	1.86E-09	sí
43	13.19757	9.342803	1.86E-09	sí	93	12.87842	8.527021	1.86E-09	sí
44	12.10916	7.687233	1.86E-09	sí	94	12.1893	8.216643	1.86E-09	sí
45	12.43834	7.448777	1.86E-09	sí	95	11.03698	7.155322	1.86E-09	sí
46	12.46958	8.23738	1.86E-09	sí	96	10.78906	5.924051	1.86E-09	sí
47	11.40021	6.14318	1.86E-09	sí	97	13.99673	9.417046	1.86E-09	sí
48	10.68896	6.647541	1.86E-09	sí	98	12.57431	7.259514	1.86E-09	sí
49	12.62583	8.009528	1.86E-09	sí	99	11.14225	6.186337	1.86E-09	sí
50	11.57072	7.390082	1.86E-09	sí	100	12.39527	8.173493	1.86E-09	sí

Tabla A.9: Prueba de *Wilcoxon* con las instancias de tamaño 200

Bibliografía

[1] Casey, R., MeMillen, K., Reynolds, R., Spaw, J. M., Schwarzer, R., Gevirtz, J., and Bauer, M. (1979). Relict and expatriated radiolarian fauna in the Gulf of Mexico and its implications. GCAGS Transactions.

[2] Conrad, J. M., Gomes, C. P., van Hoeve, W.-J., Sabharwal, A., and Suter, J. F. (2012). Wildlife corridors as a connected subgraph problem. *Journal of Environmental Economics and Management*, 63(1):1–18.

[3] Cortés Riveral, D., Landa Becerra, R., and Coello Coello, C. A. (2007). Cultural algorithms, an alternative heuristic to solve the job shop scheduling problem. *Engineering Optimization*, 39(1):69–85.

[4] CPLEX, I. I. (2009). V12. 1: User's manual for CPLEX. *International Business Machines Corporation*, 46(53):157.

[5] Deb, K. (2000). An efficient constraint handling method for genetic algorithms. *Computer methods in applied mechanics and engineering*, 186(2):311–338.

[6] Eberhart, R. and Kennedy, J. (1995). A new optimizer using particle swarm theory. In *Micro Machine and Human Science, 1995. MHS'95., Proceedings of the Sixth International Symposium on*, pages 39–43. IEEE.

[7] Eiben, A. E., Smith, J. E., et al. (2003). *Introduction to evolutionary computing*, volume 53. Springer.

[8] Fogel, L. J., Owens, A. J., and Walsh, M. J. (1966). *Artificial intelligence through simulated evolution*. John Wiley.

[9] Garfinkel, R. S. and Nemhauser, G. L. (1972). *Integer programming*, volume 4. Wiley New York.

[10] Goldberg, D. E. and Holland, J. H. (1988). Genetic algorithms and machine learning. *Machine learning*, 3(2):95–99.

[11] Grimaldi, R. P. (1998). *Matemáticas discreta y combinatoria: introducción y aplicaciones*. Pearson Educación.

[12] Haldar, V. and Chakraborty, N. (2015). Power loss minimization by optimal capacitor placement in radial distribution system using modified cultural algorithm. *International Transactions on Electrical Energy Systems*, 25(1):54–71.

[13] Helfert, M., Krempels, K.-H., Klein, C., Donnellan, B., and Gusikhin, O. (2016). *Smart Cities, Green Technologies, and Intelligent Transport Systems: 4th International Conference, SMARTGREENS 2015, and 1st International Conference VEHITS 2015, Lisbon, Portugal, May 20-22, 2015, Revised Selected Papers*, volume 579. Springer.

[14] Holland, J. H. (1962). *Concerning efficient adaptive systems*, volume 230. Spartan Books.

[15] Iacoban, R., Reynolds, R. G., and Brewster, J. (2003). Cultural swarms: modeling the impact of culture on social interaction and problem solving. In *Swarm Intelligence Symposium, 2003. SIS'03. Proceedings of the 2003 IEEE*, pages 205–211. IEEE.

[16] Islam, M., Shareef, H., Mohamed, A., et al. (2016). Optimal siting and sizing of rapid charging station for electric vehicles considering bangi city road network in malaysia. *Turkish Journal of Electrical Engineering & Computer Sciences*, 24(5).

[17] Islam, M. M., Shareef, H., and Mohamed, A. (2015). A review of techniques for optimal placement and sizing of electric vehicle charging stations. *Przeglad Elektrotechniczny*, 91(8):122–126.

[18] Italiano, G. F., Laura, L., and Santaroni, F. (2012). Finding strong bridges and strong articulation points in linear time. *Theoretical Computer Science*, 447:74–84.

[19] Jebari, K. and Madiafi, M. (2013). Selection methods for genetic algorithms. *International Journal of Emerging Sciences*, 3(4):333–344.

[20] Jia, L., Hu, Z., Song, Y., and Luo, Z. (2012). Optimal siting and sizing of electric vehicle charging stations. In *Electric Vehicle Conference (IEVC), 2012 IEEE International*, pages 1–6. IEEE.

[21] Jiménez, F. and Verdegay, J. L. (1999). Evolutionary techniques for constrained optimization problems. *In 7th European Congress on Intelligent Techniques and Soft Computing (EUFIT'99)*.

[22] Lam, A. Y., Leung, Y.-W., and Chu, X. (2014). Electric vehicle charging station placement: Formulation, complexity, and solutions. *Smart Grid, IEEE Transactions on*, 5(6):2846–2856.

[23] Landa Becerra, R. (2002). Algoritmos culturales aplicados a optimización con restricciones y optimización multiobjetivo. *México DF: Centro de Investigación y de Estudios Avanzados del IPN, Departamento de Ingeniería Eléctrica Sección de Computación*.

[24] Landa Becerra, R. and Coello Coello, C. A. (2004). A cultural algorithm with differential evolution to solve constrained optimization problems. In *Ibero-American Conference on Artificial Intelligence*, pages 881–890. Springer.

[25] Landa Becerra, R. and Coello Coello, C. A. (2005). A cultural algorithm for solving the job shop scheduling problem. In *Knowledge Incorporation in Evolutionary Computation*, pages 37–55. Springer.

[26] Liu, W.-Y. and Lin, C.-C. (2015). Spatial forest resource planning using a cultural algorithm with problem-specific information. *Environmental Modelling & Software*, 71:126–137.

[27] Liu, Z.-f., Zhang, W., Ji, X., and Li, K. (2012). Optimal planning of charging station for electric vehicle based on particle swarm optimization. In *Innovative Smart Grid Technologies-Asia (ISGT Asia), 2012 IEEE*, pages 1–5. IEEE.

[28] Lofberg, J. (2004). YALMIP: A toolbox for modeling and optimization in MATLAB. In *Computer Aided Control Systems Design, 2004 IEEE International Symposium on*, pages 284–289. IEEE.

[29] Madoliat, R., Khanmirza, E., and Pourfard, A. (2017). Application of pso and cultural algorithms for transient analysis of natural gas pipeline. *Journal of Petroleum Science and Engineering*, 149:504–514.

[30] Mehar, S. and Senouci, S. M. (2013). An optimization location scheme for electric charging stations. In *Smart Communications in Network Technologies (SaCoNeT), 2013 International Conference on*, volume 1, pages 1–5. IEEE.

[31] Netto, P. O. (2003). *Grafos: teoria, modelos, algoritmos*. Edgard Blücher.

[32] Ochoa Zezzatti, A., Pérez Rodríguez, R., and Cruz Acevez, I. (2016). Container ship stowage problem to humanitarian aid in palestine using cultural algorithms. *International Journal of Combinatorial Optimization Problems and Informatics*, 7(1):20.

[33] Rao, S. S. and Rao, S. S. (2009). *Engineering optimization: theory and practice*. John Wiley & Sons.

[34] Reynolds, R. G. (1994). An introduction to cultural algorithms. In *Proceedings of the third annual conference on evolutionary programming*, pages 131–139. Singapore.

[35] Reynolds, R. G. (1999). New ideas in optimization. chapter Cultural Algorithms: Theory and Applications, pages 367–378. McGraw-Hill Ltd., UK, Maidenhead, UK, England.

[36] Reynolds, R. G., Michalewicz, Z., and Cavaretta, M. J. (1995). Using cultural algorithms for constraint handling in genocop. In *Evolutionary programming*, pages 289–305.

[37] Reynolds, R. G. and Sverdlik, W. (1995). An evolution-based approach to program understanding using cultural algorithms. *International Journal of Software Engineering and Knowledge Engineering*, 5(02):211–226.

[38] Runarsson, T. P. and Yao, X. (2000). Stochastic ranking for constrained evolutionary optimization. *IEEE Transactions on evolutionary computation*, 4(3):284–294.

[39] Santana Quintero, L. V. and Coello Coello, C. A. (2006). Una introducción a la computación evolutiva y alguna de sus aplicaciones en economía y finanzas. *Revista de métodos cuantitativos para la economía y la empresa*, (2):3–26.

[40] Schaffer, J. (1989). A study of control parameters affecting online performance of genetic algorithms for function optimization. *San Meteo, California*.

[41] Schwefel, H.-P. (1977). *Numerische optimierung von computer-modellen mittels der evolutionsstrategie*, volume 1. Birkhäuser, Basel Switzerland.

[42] Schwefel, H.-P. (1981). *Numerical optimization of computer models*. John Wiley & Sons, Inc.

[43] Schwefel, H.-P. (1995). Evolution and optimum seeking. Sixth-generation computer technology series. Wiley, New York.

[44] Shi, Y. and Eberhart, R. (1998). A modified particle swarm optimizer. In *Evolutionary Computation Proceedings, 1998. IEEE World Congress on Computational Intelligence., The 1998 IEEE International Conference on*, pages 69–73. IEEE.

[45] Smith, J. C. and Taskin, Z. C. (2008). A tutorial guide to mixed-integer programming models and solution techniques. *Optimization in Medicine and Biology*, pages 521–548.

[46] Soza, C., Landa Becerra, R., Riff, M. C., and Coello Coello, C. A. (2011). Solving timetabling problems using a cultural algorithm. *Applied Soft Computing*, 11(1):337–344.

[47] Tahara, H., Urasaki, N., Senjyu, T., and Funabashi, T. (2016). Ev charging station using renewable energy. In *2016 IEEE First International Conference on Control, Measurement and Instrumentation (CMI)*, pages 48–52. IEEE.

[48] Talbi, E.-G. (2009). *Metaheuristics: from design to implementation*, volume 74. John Wiley & Sons.

[49] Vent, W. (1975). Rechenberg, ingo, evolutionsstrategie—optimierung technischer systeme nach prinzipien der biologischen evolution. 170 s. mit 36 abb. frommann-holzboog-verlag. stuttgart 1973. broschiert. *Feddes Repertorium*, 86(5):337–337.

[50] Wang, L.-T., Chang, Y.-W., and Cheng, K.-T. T. (2009). *Electronic design automation: synthesis, verification, and test*. Morgan Kaufmann.

[51] Wang, W., Song, Y., Xue, Y., Jin, H., Hou, J., and Zhao, M. (2015). An optimal vibration control strategy for a vehicle's active suspension based on improved cultural algorithm. *Applied Soft Computing*, 28:167–174.

[52] Xiong, Y., Gan, J., An, B., Miao, C., and Bazzan, A. L. (2015). Optimal electric vehicle charging station placement. In *Proceedings of the 24th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 2662–2668.

[53] Yan, X., Song, T., and Wu, Q. (2017). An improved cultural algorithm and its application in image matching. *Multimedia Tools and Applications*, pages 1–18.

[54] Yan, X., Wu, Q., and Liu, H. (2015). Digital circuit optimization design algorithm based on cultural evolution. *Metallurgical & Mining Industry*, (9).

[55] You, P.-S. and Hsieh, Y.-C. (2014). A hybrid heuristic approach to the problem of the location of vehicle charging stations. *Computers & Industrial Engineering*, 70:195–204.