Ingeniería y Ciencia
ISSN:1794-9165 | ISSN-e: 2256-4314
ing. cienc., vol. 12, no. 23, pp. 25–46, enero-junio. 2016.
http://www.eafit.edu.co/ingciencia

Algoritmo memético con operadores de inteligencia artificial para el CARP con inicio y fin no determinado y bi-objetivo

B. J. Macías [1] y C. A. Amaya [2]

Recepción: 24-08-2015 | Aceptación: 01-12-2015 | En línea: 01-02-2016

MSC: 90C08, 90C35, 90C29, 90C59, 68T20

doi:10.17230/ingciencia.12.23.2

Resumen

El Problema de ruteo de vehículos sobre arcos con punto de inicio/fin variable (Open Capacitated Arc Routing Problem - OCARP), en su versión clásica, busca determinar la mejor estrategia para servir un conjunto de clientes localizados en los arcos de una red usando vehículos. A diferencia del Capacitated Arc Routing Problem (CARP), el OCARP no tiene las restricciones que aseguran que cada vehículo debe iniciar y terminar su ruta en un vértice dado (también conocido como depósito). El objetivo de este trabajo es proponer una heurística para encontrar la frontera eficiente dados dos objetivos: minimizar el número de vehículos y minimizar el costo total. Adicionalmente se propone complementar la heurística, la cual es basada en algoritmos genéticos, con operadores de inteligencia artificial.

Palabras clave: algoritmo genético; algoritmo memético; optimización multiobjetivo; CARP; OCARP; MO-OCARP; redes neuronales; búsqueda local; ruteo de vehículos sobre arcos.

[1] Universidad de los Andes, Bogotá, Colombia, bj.macias126@uniandes.edu.co.
[2] Universidad de los Andes, Bogotá, Colombia, ca.amaya@uniandes.edu.co.

Hybrid Algorithm Enhanced with Artificial Intelligence Applied to the Bi-Objective Open Capacitated Arc Routing Problem

Abstract
The arc routing problem with a variable starting/ending position (Open Capacitated Arc Routing Problem - OCARP), in its classic version, pursues the best strategy to serve a set of customers located in the network arcs using vehicles. Compared to the Capacitated Arc Routing Problem (CARP), the OCARP lacks of constrains that guarantee that each vehicle ought to start and end the tour at a given vertex (also known as a depot). The aim of this paper is to propose a heuristic to find an efficient frontier for the main objective functions: minimize the number of vehicles and the total cost. Additionally, a hybrid algorithm that complements the genetic algorithm with artificial intelligence operator is proposed.

Key words: genetic algorithm; memetic algorithm; multi-objective optimization; CARP; OCARP; MO-OCARP; neural networks; local search.

1 Introducción

El problema de ruteo de vehículos sobre arcos (CARP, por sus siglas en inglés) busca determinar qué vehículo y en qué secuencia se debe visitar un conjunto de arcos, definidos sobre un grafo, a los cuales se busca prestar algún servicio, minimizando una o varias funciones de costo. Este trabajo considera la situación en la cual los vehículos no necesariamente salen (o llegan) del depósito y tiene libertad de escoger el punto de inicio (o terminación) de acuerdo a decisiones bi-objetivo. Debido a que la complejidad del problema crece exponencialmente según el tamaño del grafo, en este documento se presenta un método de solución basado en Algoritmos Genéticos (AG) en el que se utiliza algoritmos de inteligencia artificial para mejorar la evolución de los individuos.

La estructura de este artículo es la siguiente: en la sección 2 se establece cuál es el problema y cómo ha sido el trabajo previo desarrollado por otros autores. En la sección 3 se describe formalmente en problema. En la sección 4 discuten las estrategias de solución trabajadas. Y finalmente, en las secciones 5 y 6 se describe el proceso de experimentación y análisis de resultados.

2 Revisión de la literatura

El CARP fue propuesto en 1981 [1] como un problema de optimización combinatoria, en el que dado un grafo *G(V,A)* no dirigido se busca determinar un conjunto de rutas que deben seguir un número fijo de vehículos de tal manera que se minimice la distancia total recorrida. Los arcos pueden ser clasificados en dos: *arcos obligatorios y arcos de paso*. Los arcos obligatorios son aquellos que tienen una demanda que debe ser satisfecha y por tanto deben ser visitados, mientras que los arcos de paso son visitados como medio de conexión entre otros dos arcos obligatorios. Todos los arcos cuentan con una distancia la cual debe ser recorrida cuando es visitado por un vehículo. Por otra parte, los vehículos son considerados iguales y tienen una capacidad máxima. Finalmente todas las rutas establecidas deben empezar y terminar en el mismo vértice llamado *depósito.*

El CARP es NP-Hard, es decir, en términos prácticos, la complejidad de resolverlo por métodos exactos crece exponencialmente de acuerdo al tamaño de las instancias (Número de nodos y arcos). Por tal razón el problema ha sido trabajado ampliamente con el uso de heurísticas y metaheurísticas. Algunas heurísticas que han obtenido buenos resultados son: Path-Scanning [2], augment-merge[1], y augment-insert [3]. Otros autores han aplicado meta-heurísticas con aún mejores resultados. Algunos ejemplos interesantes pueden ser un algoritmo basado en Búsqueda Tabú [4] y un algoritmo memético[5]. En el artículo [6] se detalla de forma más explícita las soluciones propuestas para el CARP y sus variaciones.

Como una extensión del CARP surge el OCARP (Open Capacitated Arc Routing Problem) en el que la restricción de que todas las rutas deben empezar y terminar en el depósito es omitida. Algunas aplicaciones de este problema pueden ser la determinación de la ruta que debe seguir la persona que registra el consumo de agua o luz (Lector de registros), la determinación de patrones de corte [7], el transporte de estudiantes a sus escuelas, en la que el primer niñ a transportar puede definir el inicio de la ruta, Etc.

Para solucionar el problema OCARP se han propuesto diferentes métodos. Existen propuestas de métodos exactos basados en programación matemática [8], pero por ser un problema NP-Hard algunos trabajos se han enfocado en proponer cotas usando algoritmos de programación matemática [9], sin embargo, los métodos se han enfocado el uso de metaheu-

rísticas. Dentro de esos métodos se han empleado métodos como GRASP con PATH-RELINKING [10] y algoritmos meméticos [11].

En el caso del CARP, el caso multi-objetivo se ha estudiado en menor cantidad que el mono-objetivo. La mayor parte de los métodos han sido basados en metaheurísticas, también se han usado métodos basados en math-heurísticas [12]. Las metaheurísticas con mejores resultados sobre la familia de problemas CARP son los Algoritmos Genéticos, e incluso se han propuesto Algoritmos Genéticos que minimiza tanto la distancia total de ruteo, como la longitud de la ruta más larga (Makespan) [7] mediante el framework NSGA-II *(Non-dominated sorting algorithm)* [13], este método es un AG elitista puro en donde solamente soluciones no dominadas participan en los operadores de cruzamiento y selección. Su diseño busca clasificar o ranquear los individuos según el grado de dominancia sobre otros individuos, denominado dominancia de Pareto.

A diferencia de los algoritmos de optimización con un sólo objetivo, la calidad de un algoritmo de optimización multi-objetivo no puede ser hallada comparando la mejor solución encontrada contra el óptimo verdadero del problema. Pro esta razón se han compilado una serie de métricas para comparar algoritmos de optimización multi-objetivo [14]. En general, se considera que un algoritmo es mejor que otro si la frontera de eficiencia encontrada está más cerca de la verdadera frontera de Pareto y al mismo tiempo, las soluciones están distribuidas dentro del conjunto de valores que pueden tomar las funciones objetivo.

En la literatura se encuentran diferentes propuestas para medir la calidad de los resultados de los algoritmos multi-objetivo para el CARP. Una alternativa propuesta fue usar tres medidas de desempeño [15]: *Distancia entre Fronteras*, la cual busca medir la distancia entre las soluciones de la frontera y un punto externo de referencia; Métrica *Spread*, la cual busca medir la uniformidad de las soluciones en la frontera [13] y la métrica de *Hipervolumen*, la cual mide el espacio de solución que es dominado por los miembros de una frontera[16].

Los Algoritmos Genéticos tienen múltiples variaciones que buscan alcanzar las soluciones óptimas de forma más rápida y eficiente. Una de ellas es conocida como Algoritmos Meméticos [17], donde se conserva la estructura básica de los algoritmos genéticos pero tienen un operador adicional:

Búsqueda Local. A este nivel se tiene limitaciones en la rapidez o efectividad de los métodos usados, es así que este documento busca incorporar mecanismos de aprendizaje para realizar una búsqueda más efectiva, por ejemplo mediante el uso de algoritmos de inteligencia artificial.

Dentro del desarrollo de inteligencia artificial se han creado múltiples paradigmas de aprendizaje que procuran emular el proceso cognitivo humano por parte de un autómata. Dentro de estos paradigmas se encuentran las redes neuronales artificiales que están inspiradas en el complejo sistema neuronal humano. El uso de las redes neuronales artificiales se ha expandido paulatinamente dada la efectividad que tienen en el reconocimiento de patrones. Una red neuronal artificial se puede entender como una caja negra, en la que son ingresados algunos parámetros, y la red arroja otros como salidas.

Su funcionamiento ocurre en dos etapas: una de entrenamiento y otra de ejecución. Durante la etapa de entrenamiento son ingresadas diferentes instancias. Cada vez que es ingresada una nueva instancia, la red compara la respuesta deseada con respecto a las respuestas entregadas. Dicha comparación arroja un error que es usado para auto-calibrarse. De esta forma la red está *"Aprendiendo"* de la instancia.

En la segunda etapa, etapa de ejecución, se realiza cuando se considera que la red está entrenada y las respuestas entregadas son similares a las esperadas. En este punto se considera que la red ya es capaz de reconocer patrones y determinar, para nuevas instancias, cuáles son los valores de salida esperados. Para un mayor detalle sobre las redes neuronales, el lector puede leer el capítulo 2 de Munakata [18]. Adicionalmente, acerca de la implementación de una Red Neuronal, se desarrolló la librería FANN (Fast Artificial Neural Network) implementada en C, con todas las características y generalidades de una Red Neuronal [19].

3 Definición del problema - OCARP

La Figura 1 presenta una instancia de un problema OCARP y una solución. El problema se puede definir sobre un grafo $G = \{V, A\}$, donde $V = \{0, 1, ...n\}$ es el conjunto de vértices y A es el conjunto de aristas cuyos vértices de inicio y final pertenecen a V. Dentro de A existe un sub-

conjunto de aristas $O \subseteq A$ que se requieren servir, es decir son arcos que se deben recorrer de manera obligatoria al menos una vez. Para servir los arcos obligatorios se cuenta con un conjunto $K = \{1, 2, ...m\}$ de vehículos idénticos. Cada arco tiene una demanda d_A y un costo l_A en el que se incurre al ser cruzado por algún vehículo. Por su parte, cada vehículo tiene una capacidad Q para satisfacer la demanda d_A de las aristas visitadas.

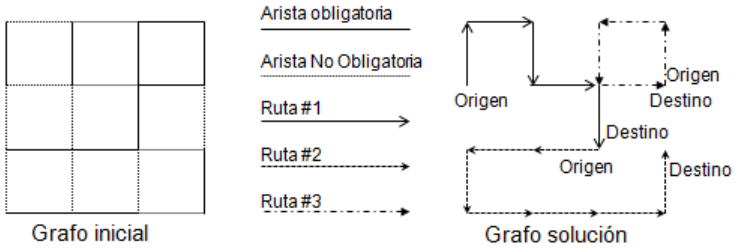

Figura 1: Ejemplo de una instancia OCARP y una solución.[1].

El objetivo es determinar un conjunto de rutas $A_R = \{a_1, a_2, ..., a_k\}$ donde cada ruta representa un conjunto ordenado de aristas $a \subseteq A$. Las rutas deben satisfacer las restricciones del problema de ruteo de vehículos sobre arcos (CARP) (ver [1]) pero sin la restricción que el vehículo salga o llegue a un punto determinado, es decir, no es obligatorio que el vehículo salga del depósito; los puntos de origen y terminación de la ruta son en sí mismo decisiones en este problema. También existe una formulación en programación lineal entera de este problema que lleva a una solución óptima del problema [8]. En este documento el problema se trabaja en una versión bi-objetivo buscando el número mínimo de vehículos que satisfagan todas las condiciones del problema y que al mismo tiempo reduzca la distancia recorrida total de todas las rutas.

4 Método de solución

De acuerdo a la literatura, los AG tienen un buen desempeño en problemas de ruteo de vehículos. El método propuesto de solución está basado en la estructura de un AG y en la utilización de un algoritmo de búsqueda local y de un algoritmo de inteligencia artificial, usando redes neurales, para

reforzar la búsqueda. La Figura 2 presenta un esquema de lo realizado en cada iteración del método propuesto.

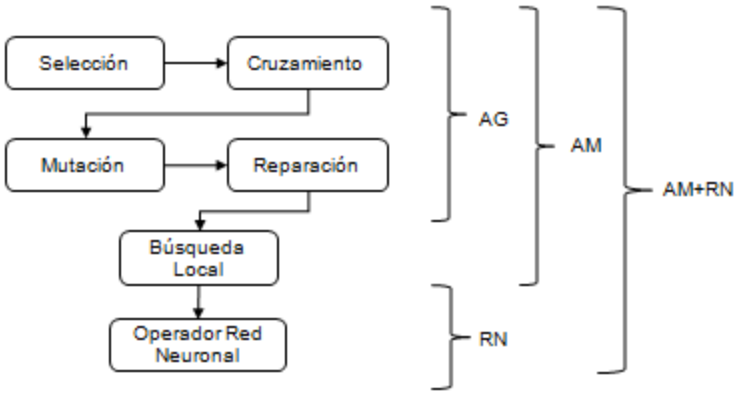

Figura 2: Esquema de cada iteración en el método propuesto
AG: Pasos clásicos de una iteración (generación) de un algoritmo genético
AM: Algoritmo Memético
RN: Operador de Red Neuronal.

El problema se modela sobre un grafo, teniendo en cuenta que los arcos de paso, arcos auxiliares, sólo son usados para comunicar otros dos arcos obligatorios, es así que la ruta formada por esos arcos de paso será manejada como un solo arco, en el cual su costo es calculado como la ruta de menor costo entre el nodo de fin de un arco obligatorio y el del comienzo del siguiente arco obligatorio. En la Figura 3 se presenta un ejemplo de grafo en el que son resaltados los arcos obligatorios.

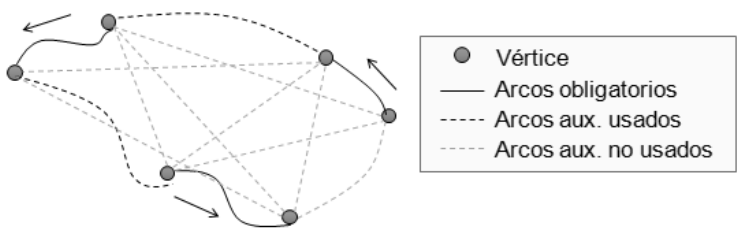

Figura 3: Grafo OCARP.

4.1 Algoritmo Genético (AG) para el problema MO-OCARP

A continuación se presenta los elementos del AG, cuya lógica es la definida ampliamente en la literatura. [13]:

4.1.1 Cromosomas y representación de una solución La Figura 4 presenta gráficamente la representación elegida, y que a su vez permite encontrar el número de vehículos usados y el orden en que son visitados los arcos:

Id Arco Obligatorio	0	1	2	3	...	k
Vehículo	0	0	1	0	...	1
Posición	3	0	0	1	...	2
Dirección	0	0	1	0	...	0

Figura 4: Representación OCARP.

Vehículo es un arreglo $\{v_i\}$ para $i \in \{Oblig\}$ donde $v_i \in \{0...MaxVehi-1\}$ indica qué vehículo tiene que visitar el arco obligatorio i. *MaxVehi* es una variable que pone cota máxima al número de vehículos del problema y toma el valor del número de arcos obligatorios.

Posición es un arreglo $\{p_i\}$ que indica en qué posición es visitado dicho arco en la ruta del vehículo correspondiente.

Dirección es un arreglo $\{t_i\}$, con t_i una variable binaria que indica si el arco obligatorio i debe ser tomado en su dirección normal ($t_i = 0$) o en dirección contraria ($t_i = 1$).

4.1.2 Métodos de selección Para el operador de selección se siguió la metodología planteada en [7] donde se define un algoritmo de torneo basado en NSGA II, en dicho torneo, dos individuos son seleccionados aleatoriamente entre toda la población y aquel que esté en una capa más cercana a la frontera de Pareto es seleccionado, si los dos individuos están en la

misma capa, se selecciona aquel individuo que esté más aislado, calculado con la medida *Crowning Distance (ver* [13])

Este proceso es ejecutado para cualquier par de individuos a los que se desee ejecutar el proceso de cruzamiento.

4.1.3 Método de cruzamiento Dada la representación ya mencionada, se decidió implementar el método de cruzamiento conocido como *Two Point Cross Over* o recombinación en dos puntos. Para ello se seleccionan dos puntos enteros aleatorios (u_1, u_2)en el rango (*1*, numArc-1); luego, el individuo hijo adquiere los cromosomas del padre desde el cromosoma 0 hasta u_1, luego de la madre desde el cromosoma u_1 hasta u_2 y finalmente, adquiere nuevamente los cromosomas del padre desde el número u_2 hasta el final.

4.1.4 Método de mutación Para el proceso de mutación son seleccionados individuos de forma aleatoria con una probabilidad considerablemente baja. Una vez se ha decidido estocásticamente que un individuo debe ser mutado se ejecuta el siguiente procedimiento: primero es escogido de forma aleatoria uno de los arcos obligatorios, y luego todas las características de dicho arco (vehículo que lo visita, posición en ser visitado y dirección) son generadas de nuevo de forma aleatoria. Para el proceso de mutación son seleccionados individuos de forma aleatoria con una probabilidad considerablemente baja. Una vez se ha decidido estocásticamente que un individuo debe ser mutado se ejecuta el siguiente procedimiento: primero es escogido de forma aleatoria uno de los arcos obligatorios, y luego todas las características de dicho arco (vehículo que lo visita, posición en ser visitado y dirección) son generadas de nuevo de forma aleatoria.

4.1.5 Método de elitismo o supervivencia El proceso de elitismo está acoplado al NSGA II de tal manera que cuando se deben seleccionar aquellos individuos que continúan en la generación se realiza buscando dejar los individuos más próximos a la frontera eficiente y que al mismo tiempo tengan funciones objetivo dispersas.

4.1.6 Operador de reparación Dada la representación usada, los operadores de cruzamiento y mutación pueden generar individuos que no cum-

plan con la restricción de capacidad de los vehículos. Cuando se está evaluando la función objetivo del costo total es necesario conocer la ruta que seguirá cada uno de los vehículos. Mientras que se está calculando dicho costo se puede verificar que el vehículo no haya copado su capacidad máxima. En el momento en el que el vehículo no tiene capacidad para visitar más arcos, el resto de la ruta que tenía asignada es ahora abastecida por un nuevo vehículo.

Esto en la representación es equivalente a reiniciar el arreglo de posiciones para empezar una nueva ruta, y cambiar el vehículo que visitaba dichos arcos. Recordemos que esto sólo se realiza a individuos infactibles a diferencia de la representación presentada en [7] y por tanto no se está favoreciendo ninguna función objetivo.

4.2 Estrategia de búsqueda local

Los algoritmos de búsqueda local funcionan bien en problemas con un solo objetivo, pero determinar si un vecino es mejor que otro, cuando se tienen múltiples objetivos no es una tarea fácil. Por esta razón en [7] se propone una función de comparación que determina cuándo un individuo es mejor para la población. La ecuación de evaluación se puede ver en la ecuación (1).

$$\begin{aligned} LS4 : mejor(S, S^{'}) = \\ w_1(f_1(S^{'}) - f_1(S)) + (1 - w_1)(f_2(S^{'}) - f_2(S)) < 0 \end{aligned} \tag{1}$$

La función $mejor(S, S^{'})$ retorna Verdadero o Falso dependiendo si el individuo S' es mejor que el individuo S. Esta función pondera la diferencia de las funciones objetivos por $w_1 \in [0, 1]$. La ecuación (2) muestra el cálculo de w_1.

$$w_1 = \left(\frac{f_1(S) - f_1^{min}}{f_1^{max} - f_1^{min}}\right) / \left(\frac{f_1(S) - f_1^{min}}{f_1^{max} - f_1^{min}} + \frac{f_2(S) - f_2^{min}}{f_2^{max} - f_2^{min}}\right) \tag{2}$$

Este cálculo ponderado fue formulado por Murata.[20] e indica un gradiente que depende de ubicación del individuo en cuestión en la frontera

actual. La Figura 5 muestra el gradiente generado que busca mejorar la frontera.

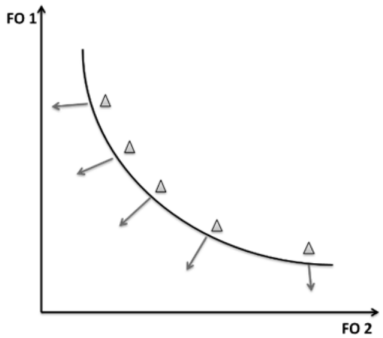

Figura 5: Operador búsqueda local.

El algoritmo de búsqueda local propuesto está inspirado con la idea de la Figura 6. Para cada arco obligatorio se evalúa cuál es el impacto en la función objetivo al cambiar la posición en la que es visitado por 1, o 2 posiciones antes o después de la configuración actual. La complejidad de evaluar el cambio de la posición de un arco obligatorio se puede simplificar a calcular el costo de los arcos auxiliares que se dejarán de usar y el costo de los arcos auxiliares que se van a requerir. De esta forma no hace falta recalcular las funciones objetivo de cada individuo, pues el número de vehículos no varía y el costo total sólo se ve afectado por el neto de cambiar los arcos auxiliares.

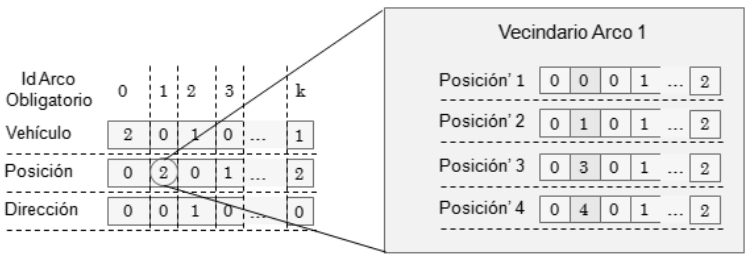

Figura 6: Operador búsqueda local.

4.3 Algoritmo híbrido: operador red neuronal artificial

En este trabajo se propone el enriquecimiento de los algoritmos genéticos con un operador que usa inteligencia artificial. Este concepto puede interpretarse como un proceso de aprendizaje dentro de la selección natural, donde los individuos pueden aprender de las mejoras que surtieron efecto en sus predecesores y se aplicarían a ellos mismos para ser competitivos.

Cada vez que se ejecuta un operador de cruzamiento, mutación o búsqueda local, se están modificando los genes o características para obtener una configuración con mejores funciones objetivo. De forma lógica, realizar la misma modificación a un conjunto de individuos no va a tener la misma eficiencia, e incluso podría dañar la solución en algunos casos. Entonces puede ser improductivo mantener un historial de modificaciones exitosas con la intención de volverlas a utilizar para mejorar futuros individuos. Pero, si tenemos en cuenta que la mejora que experimentó un individuo se debió a patrones en sus características, aplicar un cambio similar a individuos con patrones similares es muy posible que genere una mejora al individuo. La cuestión recae en el reconocimiento de dichos patrones, tarea que puede ser resuelta por una Red Neuronal Artificial.

Para la construcción de la red es necesario determinar los parámetros de entrada y de salida.

Inputs a la red:

Cada vez que un individuo experimenta una mejora, se ingresan sus genes a la red neuronal para que pueda encontrar en ellos los patrones que dieron paso a la mejora. Un ejemplo teórico de un posible patrón que puede ser reconocido es el de un vehículo que sólo visita un arco. Si dicho arco se incluyera en la ruta de otro vehículo entonces la función objetivo de reducir el número de vehículos se ve mejorada.

Para el OCARP es necesario realizar una simplificación de la representación de un individuo, pues crear una red neuronal con un número de neuronas en la capa inicial igual al número de genes de un individuo aumenta significativamente la complejidad de la misma.

Por esta razón, la representación para ingresar a la red neuronal es un arreglo $S = \{s_1, ..., s_n\}$ con $n =$ número de arcos obligatorios donde $s_i \in \{0, 1, 2, ..., n\}$ representa el siguiente arco que será visitado después de

visitar el arco obligatorio i. Se debe notar que con esta representación se está incluyendo la información del vehículo que visita cada arco y la ruta que sigue luego de visitarlo excluyendo por simplicidad la dirección en la que debía tomar el arco i.

Outputs de la red:

La respuesta esperada al ingresar un individuo es una descripción de las modificaciones que pueden ser realizadas al mismo para mejorar sus funciones objetivo. Estas modificaciones se pueden apreciar en un nuevo individuo con las modificaciones realizadas. Entonces los outputs deseados en la etapa de entrenamiento es el arreglo $S' = \{s_1^{'}, ..., s_n^{'}\}$ del individuo mejorado.

En la etapa de ejecución, se considera que la salida de la red es el individuo que ingresó como parámetros, pero con las modificaciones sugeridas a partir de información histórica.

4.3.1 Otras características de la red neuronal Adicionalmente, es necesario establecer otros parámetros para la Red Neuronal como el número de capas ocultas y el número de neuronas en cada una de ellas.

Como las instancias probadas del OCARP pueden contener hasta 190 arcos obligatorios, una Red Neuronal de ese tamaño tiene una alta complejidad. Por esta razón se deben mantener la menor cantidad de neuronas en las capas ocultas. Otro de los parámetros para la selección del número de neuronas ocultas es que debe ser mayor al número de neuronas de salida, pues de otra forma se perdería generalidad y la red no tendría los efectos esperados en nuevas instancias. Entonces basado en estos parámetros, se probaron los resultados dejando el número de neuronas ocultas en función del número de arcos del problema mediante la ecuación $num\_neuronas\_ocultas = n * \frac{4}{3}$ donde n es el Número de arcos obligatorios.

También fueron establecidos los algoritmos de *Back Propagation* para la modificación de pesos en la fase de entrenamiento, y la función de transferencia lineal para el cómputo del efecto en las neuronas. Estos fueron seleccionados dado que las variables del problema son números naturales mayores que 0.

5 Experimentación y análisis de resultados

La experimentación se realizó con el objetivo de comparar los métodos de solución antes descritos, de la siguiente manera: Algoritmo Genético*(AG)* sin búsqueda local; Algoritmo Memético *(AM)*, AG con búsqueda local definida en la sección 4.2; Algoritmo Híbrido resultado de integrar al AG y el operador con la Red Neuronal descrita en la sección 4.3 *(AG+RN)*; y el Algoritmo Híbrido integrando el Algoritmo Memético y la Red Neuronal *(AM+RN)*.

Las instancias tomadas en los experimentos provienen de adaptaciones de instancias creadas para el CARP. Dado que para el OCARP no es relevante por cuáles arcos auxiliares debe pasar un vehículo sino el orden en que visita los arcos obligatorios, en una etapa de pre-procesamiento las instancias son sintetizadas. Ahora los nodos son sólo aquellos en los que empieza o termina algún arco obligatorio, y los arcos auxiliares son las rutas más cortas entre cada par de dichos nodos. El costo asociado a los arcos es calculado con el algoritmo de *Dijkstra* [21]

Las instancias están clasificadas en 3 conjuntos: *Kshs, Gdb*, y *Eglese*. Estas fueron escogidas dado que corresponden a instancias de tamaño pequeño, mediano y grande respectivamente. La estructura y descarga de las instancias se puede realizar desde la página del autor [1].

Cada una de las instancias fue solucionada con los 4 métodos (AG, AM, AG+RN, AM+RN) registrando por cada algoritmo los individuos no dominados dados como respuesta. Adicionalmente, para las instancias del conjunto *kshs* se implementó y desarrolló un modelo entero, basado en el propuesto en [8], en el cual se propone una formulación lineal que tiene en cuenta arcos dirigidos y cuyo objetivo es el de minimizar el costo total incurrido por los vehículos. Dicha implementación se programó de forma iterativa cambiando el número de vehículos disponibles para obtener la frontera de Pareto, iterando el número de vehículos desde 0 hasta el número de arcos obligatorios. Para las demás instancias no fue posible encontrar soluciones en un tiempo razonable (se experimentó con 1 hora de tiempo, usando Xpress sobre un commputador con procesador Intel Core i5 de 2.53 GHz y 4 GB de memoria RAM), es entendible pues para esas instancias el número de variables y de restricciones crece de forma exponencial con

[1] http://www.uv.es/belengue/carp.html

respecto al tamaño de la instancia, haciendo costosa y casi imposible su solución.

Los resultados parciales obtenidos para los cuatro métodos realizados se pueden ver en las gráficas de la Figura 7. éstas corresponden a instancias escogidas por el comportamiento de las fronteras. En ellas se puede apreciar que el efecto de realizar una búsqueda local a los individuos profundiza las soluciones encontrando una frontera mejorada.

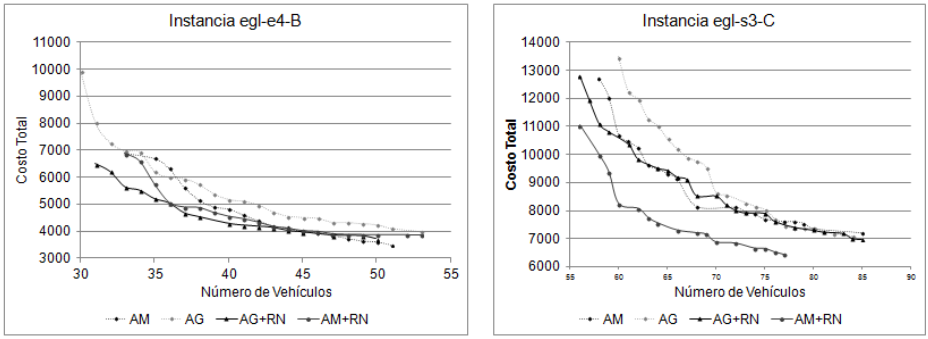

Figura 7: Fronteras mejoradas. AG: Algoritmo Genético sin búsqueda local; AM: Algoritmo Memético = AG con búsqueda local; AG + RN: Algoritmo Híbrido resultado de integrar al AG y el operador con la Red Neuronal; AM+RN: AG con búsqueda local y operador de Red Neuronal.

Por otra parte, en el conjunto de gráficas de la Figura 8 se puede interpretar la tendencia que tienen los algoritmos al implementar el operador de inteligencia artificial. En ellas, aunque no se mantiene una frontera con todos sus individuos mejores, se puede observar que las soluciones tienden a tener funciones objetivo mayormente distribuidas. Esto se considera una mejora a la frontera, pues su objetivo es servir de soporte a la toma de decisiones, y una frontera que permita ver escenarios extremos es más adecuada.

El comportamiento anteriormente mencionado, aunque sucede con frecuencia, no puede ser generalizado a todas las instancias. Esto se debe a que cada instancia contiene características que pueden favorecer un resultado. Es por esto que los métodos anteriores son definidos como heurísticos, donde por su naturaleza no exacta, existe variabilidad en los resultados

entregados.

Para obtener un criterio de comparación objetivo se implementaron las métricas *Hipervolumen* y *Spread*. La primera calcula el volumen generado por una frontera con respecto a un punto de referencia [14] y Spread busca calcular la uniformidad en las soluciones [13]:

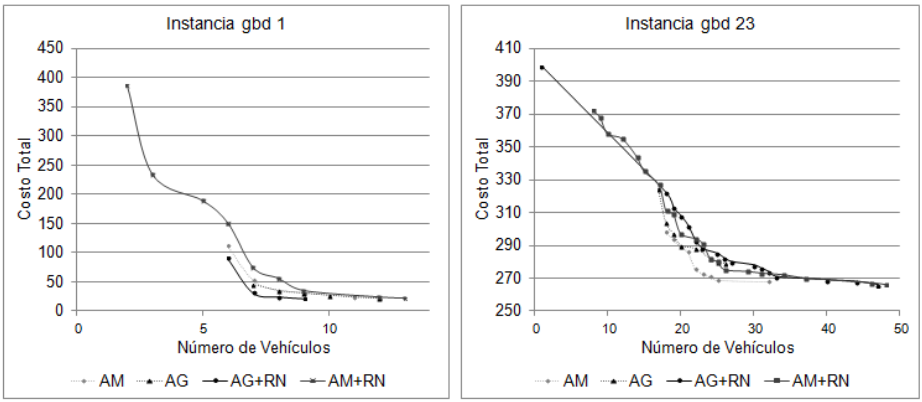

Figura 8: Fronteras diversificadas.

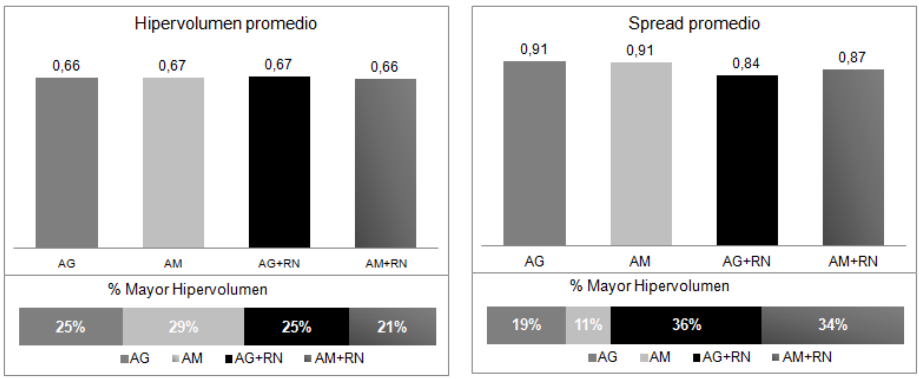

Figura 9: Métricas de comparación. (a) Hipervolumen (b) Spread.

Las gráficas de la Figura 9 muestran el valor promedio de la métrica a

través de las instancias, y el porcentaje de ocasiones en el que el algoritmo superó a los demás.

Los resultados nos muestran que no hay una diferencia significativa entre los métodos, conclusión que se puede verificar nuevamente en las gráficas de la Figura 8. en las que las fronteras de los algoritmos no presentan diferencias significativas. Sin embargo podemos ver que los algoritmos con redes neuronales tienen, aunque pequeña, una mejor dispersión de sus individuos (Menor Spread) y que el algoritmo memético tiene un mejor hipervolumen en múltiples instancias.

Adicionalmente, se calculó la dominación promedio de un algoritmo A sobre otro B midiendo el número de soluciones de B que son dominadas por A, este cálculo fue introducida en [22] y se conoce como la métrica-C. En la Figura 10 se despliega el porcentaje de individuos dominados en promedio de un algoritmo sobre otro. En la gráfica, entre mayor es el tamaño de la esfera, mayor es el porcentaje de individuos dominados.

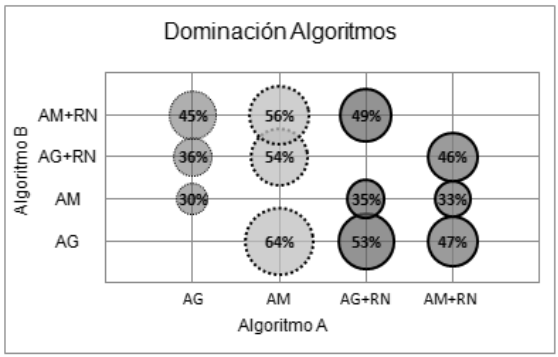

Figura 10: Dominación entre algoritmos.

Según el porcentaje de dominación, el mejor algoritmo es el memético, que como ya se mencionó profundiza los individuos de la población. Sin embargo, como se desea evaluar la dispersión de los individuos a lo largo de la frontera, también se calculó el promedio de la métrica de Amontonamiento (Crowding Distance [13]) de los individuos que no son dominados en cada algoritmo.

El procedimiento para su cálculo es el siguiente: para cada instancia se consolidan todos los individuos de las fronteras eficientes de los cuatro algoritmos. De estos, se toman aquellos que no son dominados por ningún otro individuo que corresponden a las soluciones que verdaderamente aportó cada algoritmo. Para las soluciones no dominadas se calculó la *Distancia de Amontonamiento* sin olvidar cuál es el algoritmo que la generó. Finalmente, se promedian las distancias de los individuos por algoritmo obteniendo de esta forma una métrica que indica cuál es la dispersión de los individuos aportados por cada algoritmo con respecto a los demás.

Los resultados se pueden ver en la Figura 11. Vale la pena recalcar que aquellos individuos con mayor distancia son los que se encuentran más alejados del centro.

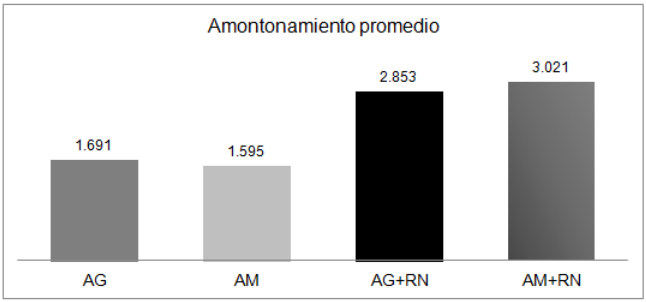

Figura 11: Amontonamiento promedio.

Con esta medida se confirma que en promedio el operador de Inteligencia Artificial permite encontrar soluciones que sobresalen de las otras fronteras.

Finalmente, en la Figura 12 se visualiza el tiempo incurrido por cada algoritmo para la resolución de 50 generaciones con instancias con diferente número de arcos obligatorios. Como es de esperarse el tiempo crece en función del tamaño de la instancia, y además se observa que al incluir la red neuronal la complejidad computacional se eleva. Esto se debe a que la red neuronal debe ser entrenada para cada instancia, proceso que requiere ajuste de cada peso de la red neuronal.

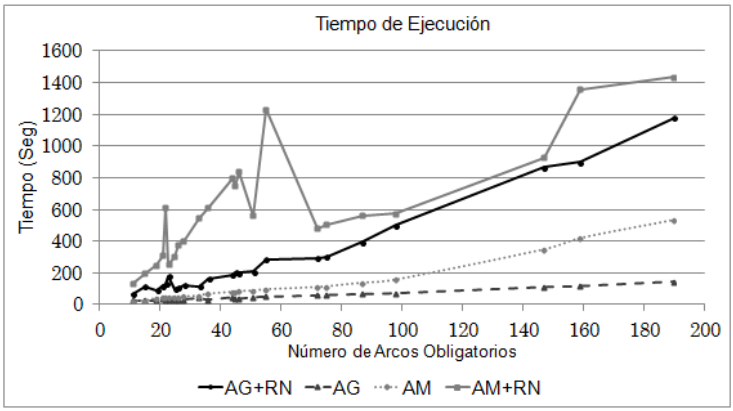

Figura 12: Tiempo ejecución algoritmos.

6 Conclusiones

Este trabajo presenta un marco general de solución para el problema de ruteo sobre arcos cuando no existe un punto fijo de inicio o terminación de la ruta, en su versión multi-objetivo - MO-CARP. El artículo propone un método de solución controlado por la lógica de un Algoritmo Genético (AG). Dentro del algoritmo se propone dos estrategias que mejoran el rendimiento del AG, la primera mediante una heurística de búsqueda local y la segunda de un mecanismo de inteligencia artificial, específicamente de una red neuronal que captura características importantes de las buenas soluciones, a ser transmitidas a las soluciones descendientes dentro del proceso evolutivo del AG. De la experimentación se encontró que no se puede establecer que una estrategia u otra puedan ser determinadas como lo mejor para la resolución del MO-CARP. Por ejemplo, mientras que el AG es capaz de construir una frontera eficiente en poco tiempo, una bśqueda local no tan extensa puede mejorar los resultados obtenidos a cambio de elevar el tiempo computacional.

Con respecto al mecanismo u operador con Redes Neuronales, su implementación aumentó la diversificación de los individuos sustrayendo eficiencia temporal para el entrenamiento de la red. Además, hay que notar que no en todas las instancias probadas ocurrieron resultados favorables. Esto

se debe a que al igual que sucede en un operador con búsqueda local, la red neuronal puede verse sesgada a óptimos locales, y guiar todos los individuos a malas soluciones. En este trabajo se propone utilizar mayor información en el proceso de generación de nuevos individuos, la Red Neuronal puede ser usada para reconocer patrones y en consecuencia determinar qué mejora podría tener buenos resultados. Consideramos que es necesario buscar mecanismos más rápidos para el aprendizaje de la Red y/u otros algoritmos de inteligencia artificial que puedan detectar más rápidamente las características que sean importantes y que se quieran transmitir a las siguientes generaciones. Esto último consideramos podría ser el análogo al proceso de manipulación genética en la creación de nuevos organismos.

Referencias

[1] B. L. Golden and R. T. Wong, "Capacitated arc routing problems," *Networks*, vol. 11, no. 3, pp. 305–315, 1981. [Online]. Available: http://dx.doi.org/10.1002/net.3230110308 27, 30

[2] L. Santos, J. Coutinho-Rodrigues, and J. R. Current, "An improved heuristic for the capacitated arc routing problem," *Computers & Operations Research*, vol. 36, no. 9, pp. 2632 – 2637, 2009. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0305054808002281 27

[3] W. L. Pearn, "Augmentinsert algorithms for the capacitated arc routing problem," *Computers & Operations Research*, vol. 18, no. 2, pp. 189 – 198, 1991. [Online]. Available: http://www.sciencedirect.com/science/article/pii/030505489190089A 27

[4] R. W. E. Leon Y. O. Li, "An interactive algorithm for vehicle routeing for winter - gritting," *The Journal of the Operational Research Society*, vol. 47, no. 2, pp. 217–228, 1996. [Online]. Available: http://www.jstor.org/stable/2584343 27

[5] P. Lacomme, C. Prins, and W. Ramdane-Cherif, "Competitive memetic algorithms for arc routing problems," *Annals of Operations Research*, vol. 131, no. 1-4, pp. 159–185, 2004. [Online]. Available: http://dx.doi.org/10.1023/B%3AANOR.0000039517.35989.6d 27

[6] A. Corberán and C. Prins, "Recent results on arc routing problems: An annotated bibliography," *Networks*, vol. 56, no. 1, pp. 50–69, 2010. [Online]. Available: http://dx.doi.org/10.1002/net.20347 27

[7] P. Lacomme, C. Prins, and M. Sevaux, "A genetic algorithm for a bi-objective capacitated arc routing problem," *Computers & Operations Research*, vol. 33, no. 12, pp. 3473 – 3493, 2006, part Special Issue: Recent Algorithmic Advances for Arc Routing Problems. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0305054805000730 27, 28, 32, 34

[8] F. L. Usberti, P. M. França, and A. L. M. França, "The open capacitated arc routing problem," *Computers & Operations Research*, vol. 38, no. 11, pp. 1543 – 1555, 2011. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0305054811000244 27, 30, 38

[9] R. Liu, Z. Jiang, N. Geng, and T. Liu, "A heuristic approach for the open capacitated arc routing problem," in *Supply Chain Management and Information Systems (SCMIS), 2010 8th International Conference on*, Oct 2010, pp. 1–6. 27

[10] F. L. Usberti, P. M. França, and A. L. M. França, "{GRASP} with evolutionary path-relinking for the capacitated arc routing problem," *Computers & Operations Research*, vol. 40, no. 12, pp. 3206 – 3217, 2013. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0305054811003029 28

[11] R. Y. Fung, R. Liu, and Z. Jiang, "A memetic algorithm for the open capacitated arc routing problem," *Transportation Research Part E: Logistics and Transportation Review*, vol. 50, pp. 53 – 67, 2013. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1366554512000932 28

[12] L. Grandinetti, F. Guerriero, D. Laganá, and O. Pisacane, "An optimization-based heuristic for the multi-objective undirected capacitated arc routing problem," *Computers & Operations Research*, vol. 39, no. 10, pp. 2300 – 2309, 2012. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0305054811003649 28

[13] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: Nsga-ii," *Evolutionary Computation, IEEE Transactions on*, vol. 6, no. 2, pp. 182–197, Apr 2002. 28, 32, 33, 40, 41

[14] E. Zitzler, L. Thiele, M. Laumanns, C. Fonseca, and V. da Fonseca, "Performance assessment of multiobjective optimizers: an analysis and review," *Evolutionary Computation, IEEE Transactions on*, vol. 7, no. 2, pp. 117–132, April 2003. 28, 40

[15] Y. Mei, K. Tang, and X. Yao, "Decomposition-based memetic algorithm for multiobjective capacitated arc routing problem," *Evolutionary Computation, IEEE Transactions on*, vol. 15, no. 2, pp. 151–165, April 2011. 28

[16] E. Zitzler, "Evolutionary algorithms for multiobjective optimization: Methods and applications," 1999. 28

[17] W. Hart, N. Krasnogor, and J. Smith, Eds., *Recent Advances in Memetic Algorithms*, 2004. 28

[18] T. Munakata, *Fundamentals of the New Artificial Intelligence: Neural, Evolutionary, Fuzzy and More (Texts in Computer Science)*, 2nd ed. Springer Publishing Company, Incorporated, 2008. 29

[19] S. Nissen, "Implementation of a Fast Artificial Neural Network Library (fann)," *Report, Department of Computer Science University of Copenhagen (DIKU)*, vol. 31, 2003. 29

[20] T. Murata, H. Nozawa, H. Ishibuchi, and M. Gen, "Modification of local search directions for non-dominated solutions in cellular multiobjective genetic algorithms for pattern classification problems," in *Evolutionary Multi-Criterion Optimization*, ser. Lecture Notes in Computer Science, C. Fonseca, P. Fleming, E. Zitzler, L. Thiele, and K. Deb, Eds. Springer Berlin Heidelberg, 2003, vol. 2632, pp. 593–607. [Online]. Available: http://dx.doi.org/10.1007/3-540-36970-8_42 34

[21] E. W. Dijkstra, "A note on two problems in connexion with graphs," *Numerische Mathematik*, vol. 1, no. 1, pp. 269–271, 1959. 38

[22] E. Zitzler and L. Thiele, "Multiobjective optimization using evolutionary algorithms - a comparative case study," in *Proceedings of the 5th International Conference on Parallel Problem Solving from Nature*, ser. PPSN V. London, UK, UK: Springer-Verlag, 1998, pp. 292–304. [Online]. Available: http://dl.acm.org/citation.cfm?id=645824.668610 41