Comparación de un sistema de colonias de hormigas y una estrategia evolutiva para un Problema Multiobjetivo de Ruteo de Vehículos con Ventanas de Tiempo.

Augusto Hermosilla y Benjamín Barán
Universidad Nacional de Asunción, Centro Nacional de Computación,
San Lorenzo, Paraguay, Casilla de Correo 1439
ahermosilla@cnc.una.py, *bbaran@cnc.una.py*

Abstract

The present work compares an Ant Colony System (ACS) with an Evolutionary Strategy (variant of the Pareto Archived Evolutionary Strategy) utilized in solving a multiobjective vehicle routing problem with time windows (VRPTW). We analyze instances of different classes and sizes, widely studied in the literature. Computational results show that the ACS has a better performance than the Evolutionary Strategy, especially in instances of larger size.

Keywords: Multiobjective Optimization, Vehicle Routing Problem with Time Windows, Ant Colony System, Pareto Archived Evolutionary Algorithm.

Resumen

El presente trabajo compara un Sistema de Optimización basado en Colonias de Hormigas (*Ant Colony Optimization*) con una estrategia evolutiva (variante del *Pareto Archived Evolutionary Strategy*), utilizados en la resolución multiobjetivo del problema de ruteo de vehículos con ventanas de tiempo. (*Vehicle Routing Problem with Time Windows*, VRPTW). Se analizan problemas de diversos tipos y tamaños, ampliamente estudiados en la literatura. Resultados experimentales demuestran que el sistema de colonias de hormigas tiene un mejor desempeño en más tipos de problemas que la estrategia evolutiva, especialmente en problemas más grandes.

Palabras claves: Optimización multiobjetivo, Problema del Ruteo de Vehículos con Ventanas de Tiempo, Sistema de Colonias de Hormigas, Pareto Archived Evolutionary Algorithm.

1. Introducción

El problema del ruteo de vehículos (*Vehicle Routing Problem*, o VRP) es ya considerado un paradigma en la literatura especializada [4]. Este problema se plantea como un depósito central que cuenta con una flota de vehículos y debe atender a un conjunto de clientes geográficamente distribuidos. El objetivo del VRP es entregar bienes a este conjunto de clientes con demandas conocidas, al mínimo coste, encontrando las rutas óptimas que se originan y terminan en el referido depósito. Cada cliente es servido una sola vez y todos los clientes deben ser atendidos, para lo cual se los asigna a vehículos que llevarán la carga (demanda de los clientes que visitará) sin exceder su capacidad máxima de transporte.

Para extender el VRP tradicional agregando la restricción adicional de asociar una ventana de tiempo a cada cliente, se define un intervalo en el que cada cliente debe ser atendido y se obtiene el problema del ruteo de vehículos con ventanas de tiempo (*Vehicle Routing Problem with Time Windows*, o VRPTW) [22]. Al considerar estas ventanas de tiempo, el costo total de ruteo y planeamiento (*scheduling*) incluyen: la distancia total recorrida que está asociada al tiempo total de viaje efectivo, el tiempo total de espera en que se incurre cuando un vehículo llega muy temprano a la ubicación de un cliente y el tiempo total de servicio (tiempo para descargar todas las mercaderías solicitadas por cada cliente). Claramente, el tiempo en el que se inicia el servicio a un cliente debe ser mayor o igual al inicio de su ventana de tiempo y el instante en que se llega a cada cliente debe ser menor o igual al fin de su ventana de tiempo. Si un vehículo llega a la ubicación de un cliente antes del inicio de su ventana de tiempo, debe esperar hasta esa hora para servir a ese cliente.

El problema espacial del ruteo de vehículos ha sido extensamente estudiado en la literatura. Una gran variedad de abordajes ha sido ya publicado para resolver este problema, utilizando implementaciones paralelas [15], estrategias híbridas combinando métodos de búsqueda local con algoritmos evolutivos [21], redes neuronales [12, 26], una heurística que utiliza información de feromonas [20] y trabajos que resuelven este problema con estrategias evolutivas para optimizar la demanda de cada vehículo además de la optimización del número de vehículos y el tiempo total de viaje [23].

La restricción de ventanas de tiempo solo ha sido considerada recientemente y varios abordajes han sido presentados para resolver el VRPTW: algoritmos paralelos con un número polinomial de procesadores [13], algoritmos genéticos [8, 16, 17, 24], recocido simulado paralelo [1, 10] y colonias múltiples de hormigas [11]. Estos trabajos analizan el problema en cuestión en un contexto mono-objetivo, aunque en algunos casos se proponen priorizaciones y/o combinaciones de objetivos, sin llegar a calcular el conjunto completo de soluciones Pareto. Estudios de las principales heurísticas y metaheurísticas pueden ser encontrados, por ejemplo, en [6] y [7].

Por su parte, el presente trabajo resuelve el problema del VRPTW en un contexto multiobjetivo, considerando dos técnicas evolutivas que han demostrado su capacidad de resolver adecuadamente problemas de similar complejidad:

(1) las optimizaciones con colonias de hormigas (ACO), en la versión conocida como MOACS-VRPTW, propuesta por Barán y Schaerer [3], a partir de una optimización del MACS-VRPTW publicado por Gambardella et al. [11]; y

(2) los algoritmos evolutivos multiobjetivos (*Multi-Objective Evolutionary Algorithms* – MOEAs) en una variante propuesta en este trabajo que daremos en llamar PAES-DRI, obtenida a partir de la propuesta ES-DRI presentada por Mester [19].

Esta comparación experimental puede ser considerada como una continuación del trabajo presentado en [2].

El resto del trabajo es organizado de la siguiente manera: la sección 2 presenta la formulación tradicional del problema, la sección 3 contiene la formulación multiobjetivo, la sección 4 resume el MOACS-VRPTW, la 5 explica el PAES-DRI, la 6 presenta las métricas de desempeño utilizadas para la comparación, mientras la sección 7 presenta resultados experimentales. Finalmente, la sección 8 presenta las conclusiones del trabajo.

2. Formulación Matemática del VRPTW

El VRPTW es un problema combinatorial que puede ser formulado matemáticamente como un grafo dirigido ***G(V,A)***. El problema considera un conjunto de vértices $V = \{v_0, v_1, \ldots, v_n\}$, donde v_0 representa el depósito, mientras que v_i *(i = 1, ..., n)* representa a cada uno de los *n* clientes (o ciudades) a ser visitados. Por otro lado, el conjunto de arcos está dado por: $A = \{(v_i, v_j) / v_i, v_j \in V; i \neq j\}$ [19].

Por su parte, $C = \{c_{ij}\} \in \Re^{(n+1)x(n+1)}$ es una matriz de distancias o costos no negativos entre cada par de vértices v_i y v_j (incluyendo depósito y clientes).

$q \in \Re^n$ es el vector de las demandas de los clientes y q_i representa la cantidad de bienes requeridos por el cliente v_i.

m representa el número de vehículos, que se asumen todos idénticos, y con una capacidad *Q* por vehículo. A cada vehículo *i* se le asignará una ruta R_i.

$[e_i, l_i]$ representa la ventana de tiempo del cliente v_i; con e_i como la hora más temprana y l_i como la hora más tardía en que se puede iniciar el servicio a dicho cliente v_i. Por su parte, δ_i representa el tiempo requerido en descargar la cantidad de mercaderías q_i en el cliente v_i.

Para este trabajo, se considera sólo el problema simétrico con $c_{ij} = c_{ji}$ para cada $(v_i, v_j) \in A$. En este caso, es común reemplazar *A* por el conjunto $E = \{(v_i, v_j) \mid v_i, v_j \in V; i < j\}$. Sin pérdida de generalidad, y por simplicidad, se asume que el tiempo t_{ij} necesario para ir de v_i a v_j es igual a la distancia c_{ij} (se asume velocidad constante e unitaria para los vehículos). El intervalo $[e_0, l_0]$ en el depósito es conocido como horizonte de planeación (*scheduling horizon*).

Definiendo b_v como el instante de inicio del servicio al cliente v, se puede escribir la condición de factibilidad que debe cumplir una ruta $R_i = \{v_0^i, v_1^i, \ldots, v_{k_i}^i, v_{k_i+1}^i\}$, donde $v_j^i \in V$, $v_0^i = v_{k_i+1}^i = 0$ (el 0 denota el depósito) y k_i es la cantidad de clientes atendidos en la ruta i:

por la capacidad del vehículo: $\sum_{j=1}^{k_i} q_j^i \leq Q. \quad i = 1, \ldots, m$ **(1)**

por las ventanas de tiempo: $e_{v_j^i} \leq b_{v_j^i} \leq l_{v_j^i}. \quad i = 1, \ldots, m; \quad j = 1, \ldots, k_i + 1$ **(2)**

Asumiendo que todo vehículo i viaja al próximo cliente v_j^i una vez que haya terminado de servir al cliente actual v_{j-1}^i, el valor de $b_{v_j^i}$ puede ser calculado recursivamente de la siguiente manera:

$$b_{v_j^i} = \max\{e_{v_j^i}; b_{v_{j-1}^i} + \delta_{v_{j-1}^i} + c_{v_{j-1}^i, v_j^i}\}, \qquad \text{con } b_0^i = e_0 \text{ y } \delta_0^i = 0. \qquad \mathbf{(3)}$$

Para este trabajo, se considera $e_0 = 0$, es decir, todos los vehículos parten del depósito en el mismo instante.

Así, el tiempo de espera en el cliente v_j^i puede ser definido como:

$$w_{v_j^i} = \max\{0; b_{v_j^i} - b_{v_{j-1}^i} - \delta_{v_{j-1}^i} - c_{v_{j-1}^i, v_j^i}\}. \qquad \mathbf{(4)}$$

El costo de la ruta R_i puede estar dado por la duración total de la misma, en cuyo caso estaría definido como:

$$T(R_i) = \sum_{j=0}^{k_i} c_{v_j^i, v_{j+1}^i} + \sum_{j=1}^{k_i} \delta_{v_j^i} + \sum_{j=0}^{k_i} w_{v_j^i}. \qquad \mathbf{(5)}$$

Tradicionalmente, el objetivo primario es la minimización del tamaño *m* de la flota de vehículos y el objetivo secundario es la minimización de la distancia total recorrida (tiempo total de viaje) o la duración total de las rutas (tiempo total de entrega), [6, 7].

3. Formulación del VRPTW como un Problema Multiobjetivo

Como ya fuera discutido, el VRPTW es tradicionalmente formulado como un problema bi-objetivo lexicográfico [25]. En contrapartida, este trabajo utiliza una formulación multiobjetivo presentada en [3]. En este contexto, la función objetivo es un vector tridimensional $F = [F_1, F_2, F_3]^T$, y ningún objetivo es más importante que los otros. Los objetivos propuestos son:

- $F_1 = m$ ... es el número de vehículos (o tamaño de la flota); el cual es una medida de la inversión inicial necesaria y el costo del mantenimiento de vehículos.
- $F_2 = \sum_{i=1}^{m} \sum_{j=0}^{k_i} c_{v_j^i, v_{j+1}^i}$ es el tiempo total de viaje (o distancia total recorrida), que representa el consumo de combustible.
- $F_3 = F_2 + \sum_{i=1}^{m} \sum_{j=1}^{k_i} \delta_{v_j^i} + \sum_{i=1}^{m} \sum_{j=0}^{k_i} w_{v_j^i}$ es el tiempo total de entrega y se relaciona con el gasto necesario en los viáticos de los conductores y el seguro de los vehículos.

donde v_j^i denota el cliente servido en el *j*-ésimo orden de la ruta R_i y k_i representa el número total de clientes servidos en dicha ruta R_i.

Otros objetivos importantes y que pueden ser implementados en trabajos futuros son:

- $F_4 = max_i \left(b^i_{v_{k_i+1}}\right)$ es la duración de la ruta más larga e indica la satisfacción de los clientes y la posibilidad de asignar los vehículos a otras tareas.
- $F_5 = \sum_{i=1}^{m} \sum_{j=0}^{k_i} w_{v_j^i}$ es el tiempo total de espera, que puede ser reducido admitiendo $b_0^i \geq e_0$ en (4). Este objetivo se relaciona con el riesgo y los viáticos adicionales en los que se incurren debido a llegar a la ubicación de un cliente antes del inicio de su ventana de tiempo.

En general un problema de optimización multiobjetivo puede formularse como:

Minimizar $[f_1(x), f_2(x), \ldots, f_K(x)]^T$

Sujeto a $x \in S$

donde K es el número de objetivos, $x = (x_1, x_2, \ldots, x_l)^T$ es un vector con l variables de decisión, y S es la región factible (o conjunto de soluciones factibles).

Para comparar dos soluciones $x, y \in S$, es frecuente utilizar el concepto de dominancia [25]. En un contexto de minimización, se dice que una solución x domina a la solución y, denotado por $x \succ y$, si:

a. x es por lo menos igual a y en cada objetivo: $F_k(x) \leq F_k(y), \forall k \in \{1, \ldots, K\}$

b. x es mejor que y en por lo menos un objetivo: $\exists k \in \{1, \ldots, K\}, \quad F_k(x) < F_k(y)$.

Dado que ningún objetivo se considera más importante que los demás, se trata de un contexto realmente multiobjetivo donde en principio, la solución puede no ser única, sino un conjunto de soluciones óptimas, no comparables entre sí, conocido como *Conjunto Pareto P*. La imagen de *P* en el contra-dominio de las funciones objetivo es conocida como Frente Pareto *FP* [25].

4. Algoritmo MOACS-VRPTW

Barán y Schaerer [3] propusieron una variación del Algoritmo de Colonias Múltiples de Hormigas para el VRPTW (*Multiple Ant Colony System for Vehicle Routing Problems with Time Windows, MACS-VRPTW*), originalmente propuesto por Gambardella et al. [11]. Esta variación fue denominada Colonia de Hormigas para

Optimización Multiobjetivo del VRPTW (*Multiobjective Ant Colony System for Vehicle Routing Problem with Time Windows, MOACS-VRPTW*). Esta metaheurística utiliza una sola colonia de hormigas para minimizar simultáneamente las tres funciones objetivo, presentadas en la sección anterior. Todas las funciones comparten los mismos rastros de feromonas. De esta manera, el conocimiento de buenas soluciones es igualmente importante para cada función objetivo. El referido MOACS-VRPTW puede ser resumido con el siguiente pseudocódigo:

Pseudocode MOACS-VRPTW

/* *Inicialización* */

Repeat /* *Ciclo principal* */

    for each ant $k \in \{1,\ldots, m\}$ **do**
        Construir una solución (ψ^k)
        if $\psi^k \in P$ /* si solución encontrada forma parte del conjunto Pareto */
            grabar ψ^k y borrar soluciones dominadas de P
        end if
    end for
    /* Actualizar feromonas τ */
    if $\tau_{0'} > \tau_0$ /* Un mejor *FP* ha sido encontrado */
        Reinicializar rastros $\{\tau\}$ con τ_0
    else
        for each $\psi^P \in P$ do
            Actualización global de feromonas
        end for
    end if

until criterio de parada ha sido satisfecho.

5. Algoritmo PAES-DRI

El presente trabajo, propone una nueva variante de MOEA que damos en llamar PAES-DRI, inspirada en el ES-DRI originalmente propuesto por Mester [18]. El ES-DRI (*Evolutionary Strategy - Dichotomy Remove-Insert*) es un algoritmo heurístico muy efectivo para determinar la solución de versiones en gran escala del VRPTW. Este algoritmo está basado en las ideas de estrategias evolutivas (1+1) y algunos operadores de mutación que trabajan con un esquema dicotómico de remover-insertar. Por su parte, el PAES (*Pareto Archived Evolutionary Strategy*) [14] es un esquema simple de evolución para problemas multiobjetivo. Este algoritmo es una estrategia evolutiva (1+1) que utiliza búsqueda local en una población de un individuo, pero usando un archivo de referencia de soluciones previamente encontradas a fin de identificar el *ranking* de dominancia aproximado de las soluciones *actual* y *candidata*. El PAES-DRI, propuesto en este trabajo, puede ser resumido con el siguiente pseudocódigo:

Pseudocode PAES-DRI

Generar una solución inicial a y agregarla al conjunto Pareto P

Repeat /* *Ciclo principal* */
    Mutar a para obtener b y evaluar b
    if $(a \succ b)$ /* si la solución a domina a b */
        Descartar b
    else if $(b \succ a)$
        $a \leftarrow b$
        Agregar a al conjunto Pareto P y eliminar las soluciones dominadas de P
    else if (algún elemento de P domina a b)
        Descartar b
    else (Determinar cual será la solución actual a y si se debe agregar b a P)
    end if

until un criterio de parada ha sido alcanzado.

Considerando que esta variante está siendo propuesta en este trabajo, se la describe en mayores detalles en las siguientes subsecciones, que están organizadas de la siguiente manera: la subsección 5.1 resume la heurística de inicialización y la 5.2 presenta el operador de mutación empleado en este MOEA.

5.1. Inicialización

La inicialización de una solución es realizada utilizando una heurística basada en la inserción más económica [5]. Las rutas son construidas una a la vez en forma secuencial. Para esto, es necesario determinar los nodos que inicialicen las rutas. Estos nodos son denominados nodos *semillas* (*seed nodes*). Se selecciona un nodo como inicio de la primera ruta y se procede iterativamente a insertar nodos en esa ruta. Cuando no se pueden realizar más inserciones factibles, se selecciona otro nodo semilla para inicializar otra ruta y se procede iterativamente hasta atender a todos los clientes

5.2. Operador de mutación

Tal como se había mencionado con anterioridad, el PAES-DRI es una variación del ES-DRI [19]. El operador de mutación utilizado en dicho trabajo genera una nueva solución removiendo e insertando β clientes de la solución actual a para obtener una solución candidata b. Este procedimiento genera nuevas rutas factibles en un proceso de dos fases que construye y mejora nuevas soluciones. El número de clientes a ser removidos de la solución actual es determinado en [19] conforme a la ecuación:

$$\beta = (0.1 + 0.5rnd)\,n\,. \qquad \textbf{(6)}$$

donde n es siempre el número total de clientes y rnd es un número generado aleatoriamente y distribuido uniformemente entre 0 y 1.

Después de remover un cliente de la solución actual se procede a insertarlo en otra posición para obtener una solución candidata factible. La inserción se realiza en forma secuencial. Para insertar el cliente v_u se determinan todos los lugares factibles de inserción y se halla para cada uno de ellos el costo de inserción de acuerdo a la función vectorial $[\Delta F_1, \Delta F_2, \Delta F_3]$. Como la función de inserción tiene tres dimensiones, se determinan todas las opciones de inserción que no son dominadas y se selecciona una al azar.

6. Métricas de desempeño

Para evaluar los resultados experimentales de los dos algoritmos arriba presentados, se seleccionaron cuatro métricas, considerando que no existe una única métrica que pueda por sí sola medir el desempeño, eficiencia y efectividad de los algoritmos evolutivos multiobjetivos [25]. Las métricas utilizadas para el presente trabajo son las siguientes:

- *Overall Non-dominated Vector Generation* (ONVG): que simplemente cuenta el número de soluciones en el Frente Pareto calculado, denotado como Y_{known}

  $$ONVG \stackrel{\Delta}{=} |Y_{known}|_c\,. \qquad \textbf{(8)}$$

  donde $|\;|_c$ denota cardinalidad. A mayor valor del ONVG, se tiene un mejor conocimiento de los detalles del Frente Pareto.

- *Generational Distance* (*GD*) representa que tan alejado se encuentra $\boldsymbol{Y}_{known}$ del Frente Pareto Óptimo denotado como $\boldsymbol{Y}_{true}$. Como $\boldsymbol{Y}_{true}$ no es conocido en teoría, debe estimarse haciendo muchas corridas de diversos algoritmos multiobjetivos para el mismo problema, escogiendo las soluciones Pareto óptimas encontradas con la unión de los resultados obtenidos con todos los experimentos. Claramente, un menor valor de GD, indica que un conjunto $\boldsymbol{Y}_{known}$ es mejor que otro. La distancia generacional se define como:

  $$GD \stackrel{\Delta}{=} \frac{\left(\sum_{i=1}^{N} d_i^2\right)^{1/2}}{N}\,. \qquad \textbf{(9)}$$

  donde N es el número de vectores en $\boldsymbol{Y}_{known}$, y d_i es la distancia euclidiana (en el espacio objetivo) entre cada vector y el miembro *más cercano* en $\boldsymbol{Y}_{true}$.

- *Coverage* (*C*): cuenta el número de soluciones del Frente Pareto de un algoritmo 1, denotado como $\boldsymbol{Y}_{known1}$, que son dominadas por alguna solución del Frente Pareto del Algoritmo 2 ($\boldsymbol{Y}_{known2}$). Lógicamente, al comparar dos algoritmos, el que tenga un menor valor de *C*, será superior.

- *Spacing (S)*: mide el rango de la variancia de los vectores vecinos en $\boldsymbol{Y}_{known}$. Esta métrica está definida como:

$$S \stackrel{\Delta}{=} \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}\left(\bar{d}-d_i\right)^2} \qquad (\mathbf{10})$$

donde $d_i = min_j\left(\left|f_1^i(x)-f_1^j(x)\right|+\left|f_2^i(x)-f_2^j(x)\right|+\left|f_3^i(x)-f_3^j(x)\right|\right);\ i,j=1,2,\ldots,N$; $\bar{d}$ es la media de todos los d_i y N es el número de vectores en $\boldsymbol{Y}_{known}$. Si el *spacing* es igual a cero, entonces los vectores están uniformemente distribuidos. De esta manera, es preferible un menor valor de *spacing*.

7. Resultados Experimentales

Los problemas de prueba forman parte de un conjunto de problemas ya clásicos y ampliamente estudiados en la literatura [22]. Se han considerado diversos tamaños del problema, a fin de poder comparar el desempeño de los algoritmos.

Se han realizado diez corridas de cada algoritmo, formando los Frentes Pareto $\boldsymbol{Y}_{MOACS}$ y $\boldsymbol{Y}_{PAES\text{-}DRI}$. Como no se conoce la solución teórica óptima del problema, se aproximó $\boldsymbol{Y}_{true}$. mediante la unión de todas las soluciones obtenidas por los investigadores, utilizando los 2 algoritmos que se están comparando, y eliminando de este conjunto las soluciones dominadas.

El número de generaciones del PAES-DRI es de 1000 y el de MOACS-VRPTW es de 3000. De esta manera pueden encontrarse tiempos similares de ejecución.

En la Tabla 1 se pueden observar los resultados experimentales para instancias de 25 clientes y se pintan en gris los mejores valores experimentales para cada métrica. Con respecto a ONVG, se puede apreciar que ningún algoritmo domina realmente al otro, dado que PAES-DRI es superior a MOACS-VRPTW en las instancias C207_25, R101_25 y RC105_25. Por su parte, MOACS-VRPTW es superior a PAES-DRI en las instancias C101_25, R210_25, RC202_25.

En cuanto a las métricas GD y C, podemos observar que PAES-DRI obtiene mejores soluciones que MOACS-VRPTW en 5 de 6 instancias. De esta forma podemos afirmar que las soluciones generadas por PAES-DRI utilizan menor cantidad de vehículos y recorren una menor distancia en menor tiempo.

Por otro lado, la distribución de las soluciones obtenidas por MOACS-VRPTW es más uniforme que las generadas por PAES-DRI (ver métrica S). Pero este resultado sólo indica que MOACS-VRPTW obtiene soluciones más diversas que PAES-DRI.

Ahora podemos apreciar que PAES-DRI tiene un desempeño superior a MOACS-VRPTW en las instancias C101_25, C207_25, R101_25 y RC105_25, considerando simultáneamente las cuatro métricas.

En la Tabla 2 se pueden observar los resultados experimentales para instancias de 50 clientes. MOACS-VRPTW obtiene una mayor cantidad de soluciones no dominadas que PAES-DRI (ver métrica ONVG) en 4 de 6 instancias. Con respecto a las métricas GD y C, MOACS-VRPTW obtiene mejores soluciones que PAES-DRI en 4 de 6 instancias. Por otro lado, ningún algoritmo domina al otro en la generación de soluciones uniformemente distribuidas, ya que MOACS-VRPTW es superior en las instancias C101_50, R210_50 y RC105_50. Así mismo, PAES-DRI es superior en las instancias C207_50, R101_50 y RC202_50.

Si consideremos simultáneamente las cuatro métricas, se puede observar que MOACS-VRPTW tiene un desempeño superior a PAES-DRI en las instancias C101_50, C207_50, R210_50 y RC202_50.

Tabla 1. Métricas de desempeño para los dos algoritmos comparados en el presente trabajo en instancias de 25 clientes.

Instancia	MOEA	ONVG	GD	C	S
C101_25	MOACS	8	0.1045	2	0.1539
	PAES-DRI	5	0.0000	8	0.1610
C207_25	MOACS	11	0.3830	0	0.1180
	PAES-DRI	20	0.0000	11	0.0486
R101_25	MOACS	6	0.1329	2	0.1077
	PAES-DRI	8	0.0000	6	0.1617
R210_25	MOACS	14	0.0866	0	0.1375
	PAES-DRI	6	0.0000	6	0.3810
RC105_25	MOACS	2	0.9664	0	0.0000
	PAES-DRI	10	0.0000	2	0.1009
RC202_25	MOACS	22	0.0353	15	0.0417
	PAES-DRI	17	0.3584	3	0.0819

Tabla 2. Métricas de desempeño para los dos algoritmos comparados en el presente trabajo en instancias de 50 clientes.

Instancia	MOEA	ONVG	GD	C	S
C101_50	MOACS	5	0.0000	1	0.2805
	PAES-DRI	1	0.5840	0	0.0000
C207_50	MOACS	18	0.0000	18	0.1281
	PAES-DRI	18	0.2961	0	0.0739
R101_50	MOACS	8	0.5701	0	0.1466
	PAES-DRI	7	0.0000	8	0.1345
R210_50	MOACS	16	0.0000	4	0.0606
	PAES-DRI	4	0.1683	0	0.1869
RC105_50	MOACS	2	0.0000	0	0.0000
	PAES-DRI	3	0.0000	0	0.4750
RC202_50	MOACS	22	0.0000	13	0.1250
	PAES-DRI	19	0.2661	0	0.0858

En la Tabla 3 se pueden observar los resultados experimentales para instancias de 100 clientes. PAES-DRI obtiene una mayor cantidad de soluciones no dominadas que MOACS-VRPTW (ver métrica ONVG) en 5 de 6 instancias. En cuanto a las métricas GD y C, MOACS-VRPTW obtiene en general mejores soluciones que PAES-DRI en 4 de 6 instancias. Así mismo, podemos observar que MOACS-VRPTW es ligeramente superior a PAES-DRI en la generación de soluciones uniformemente distribuidas (ver métrica S).

De igual manera a lo procedido anteriormente, podemos considerar simultáneamente las cuatro métricas y afirmar que MOACS-VRPTW tiene un desempeño superior a PAES-DRI en las instancias C101_100, C207_100, R101_100, R210_100 y RC202_100.

Si observamos el desempeño global de los algoritmos, podemos resaltar que el MOACS-VRPTW obtiene mejores resultados en las instancias de tipo C1, C2, R2, RC2 y el PAES-DRI en R1 y RC1. Esto sugiere la necesidad de implementar una heurística en el PAES-DRI que optimice localmente cada ruta, a fin de mejorar su convergencia. Este hecho es resaltado en [9], dado que los problemas de clase C2, R2 y RC2 se caracterizan por tener soluciones en que el número de vehículos es pequeño (2-4 vehículos) y la cantidad de clientes servidos en cada ruta es relativamente grande (alrededor de 30 clientes atendidos por ruta).

Tabla 3. Métricas de desempeño para los dos algoritmos comparados en el presente trabajo en instancias de 100 clientes.

Instancia	MOEA	ONVG	GD	C	S
C101_100	MOACS	1	0.0000	2	0.0000
	PAES-DRI	2	1.2961	0	0.0000
C207_100	MOACS	1	0.0000	15	0.0000
	PAES-DRI	15	1.4841	0	0.0653
R101_100	MOACS	14	0.4224	0	0.0926
	PAES-DRI	7	0.0000	14	0.2010
R210_100	MOACS	7	0.0000	19	0.3528
	PAES-DRI	19	0.6004	0	0.0761
RC105_100	MOACS	3	0.3767	0	1.3818
	PAES-DRI	4	0.0000	1	0.2916
RC202_100	MOACS	10	0.0000	10	0.1657
	PAES-DRI	16	0.1291	0	0.4165

8. Conclusión

El problema del ruteo de vehículos con ventanas de tiempo va ganando importancia en la medida que el mundo globalizado fuerza a las empresas distribuidoras modernas a ser cada vez más eficientes e incrementar su región de trabajo, para mantenerse competitivas. Como natural consecuencia, crece el interés en el área y se van proponiendo nuevos abordajes, entre los que se destacan los algoritmos evolutivos, por su flexibilidad para analizar circunstancias cambiantes y objetivos contradictorios. A raíz de esto, resulta interesante comparar las nuevas alternativas que se van proponiendo para lograr aplicar el abordaje más adecuado a cada realidad.

Es en este contexto que el presente trabajo propone una variante de algoritmo evolutivo multiobjetivo y realiza una comparación entre un sistema de colonias de hormigas y una estrategia evolutiva para el problema del ruteo multiobjetivo de vehículos con ventanas de tiempo.

Los experimentos demuestran que el MOACS-VRPTW tiene un mejor desempeño en más tipos de problemas, pero es necesaria la implementación de una heurística en el PAES-DRI que optimice localmente cada ruta, a fin de mejorar la calidad de las soluciones a los problemas de clase C2, R2 y RC2.

Los autores proponen diversos trabajos futuros, tales como: una comparación con los mejores algoritmos mono-objetivos publicados; implementar otras variantes de algoritmos multiobjetivos utilizando manejo de restricciones y resolver otras variantes del problema de ruteo de vehículos considerando objetivos adicionales, como los propuestos en la sección 3.

Referencias

[1] Arbelaitz, O., Rodriguez, C. y Zamkola, I. Low cost parallel solutions for the VRPTW optimization problem. *Fourth International Conference on Parallel Processing Workshops*, 2001, pp.176-181.

[2] Barán B. y Hermosilla, A. Comparación de un Sistema de Colonias de Hormigas y una Estrategia Evolutiva para el Problema del Ruteo de Vehículos con Ventanas de Tiempo en un Contexto Multiobjetivo. *IX Jornadas Iberoamericanas de Informática*, Cartagena de Indias, Colombia. 2003.

[3] Barán, B. y Schaerer, M. A multiobjective Ant Colony System for Vehicle Routing Problem with Time Windows. *Proceedings of the 21st IASTED International Conference APPLIED INFORMATICS.* Innsbruck, Austria. 2003.

[4] Bodin, L. Golden, B., Assad, A. y Ball, M. Routing and Scheduling of Vehicles and Crews. The State of the Art. *Computers and Operations Research.*Vol. 10, 1983, pp.62-212.

[5] Bräysy, O. A reactive variable neighborhood for the Vehicle Routing Problem with Time Windows. 2001. To appear in *Informs Journal on Computing*.

[6] Bräysy, O. y Gendreau, M. Route construction and local search algorithms for the vehicle routing problem with

time windows, Report STF42 A01024, App. Mathematics, Dep. of Optimization, Norway. 2001.

[7] Bräysy, O. y Gendreau, M. Metaheuristics for the vehicle routing problem with time windows, Internal Report STF42 A01025, SINTEF Applied Mathematics, Department of Optimization, Norway. 2001.

[8] Chin, A., Kit, H. y Lim, A. A new GA approach for the vehicle routing problem. *11th IEEE International Conference on Tools with Artificial Intelligence*, 1999, pp.307-310.

[9] Cordeau, J.-F., Laporte, G. y Mercier, A. A Unified Tabu Search Heuristic for Vehicle Routing Problems with Time Windows. *Journal of the Operational Research Society*. Vol. 52, 2001, pp.928-936.

[10] Czech, Z.J. y Czarnas, P. Parallel simulated annealing for the vehicle routing problem with time windows. *10th Euromicro Workshop on Parallel, Distributed and Network-based Proceedings*, 2002, pp.376-383.

[11] Gambardella, L., Taillard, E. y Agazzi, G. *News ideas in optimization*. Mac Graw-Hill: London, 1999, pp.73-76.

[12] Gomes,L. y von Zuben, F.J. A neuro-fuzzy approach to the capacitated vehicle routing problem. *International Joint Conference on Neural Networks*. Vol. 2, 2002, pp.1930-1935.

[13] Gupta, A. y Krishnamurti, R. Parallel algorithms for vehicle routing problems. *Fourth International Conference on High-Performance Computing*, 1997, pp.144-151.

[14] Knowles, J. and Corne, D. The Pareto Archived Evolution Strategy: A New Baseline Algorithm for Pareto Multiobjective Optimisation. Dept. of Computer Science. University of Reading. UK. 1999.

[15] Lau, K.K., Kumar, M.J. y Achuthan, N.R. Parallel implementation of branch and bound algorithm for solving vehicle routing problem on NOWs. *3rd Internationl Symposium on Parallel Architectures, Algorithms, and Networks*, 1997, pp.247-253.

[16] Louis, S.J., Xiangying, Y. y Zhen, Y.Y. Multiple vehicle routing with time windows using genetic algorithms. *Congress on Evolutionary Computation*, 1808(3), 1999.

[17] Maeda, O., Nakamura, M., Ombuki, B.M. y Onaga, K. A genetic algorithm approach to vehicle routing problem with time deadlines in geographical information systems. *International Conference on Systems, Man, and Cybernetics*. Vol. 4, 1999, pp.595-600.

[18] Mester, D. The Parallel algorithm for Vehicle Routing Problem with Time Windows restrictions. Scientific Report, Minerva Optimization Center, Technion, Israel. 1999.

[19] Mester, D. An evolutionary strategies algorithm for large scale vehicle routing problem with capacitate and time windows restrictions, Institute of Evolution, Mathematical and Population Genetics, Univ. of Haifa, Israel. 2002.

[20] Murao, H., Tohmata, K., Konishi, M. y Kitamura, S. Pheromone based transportation scheduling system for the multi-vehicle routing problem *IEEE Int. Conference on Systems, Man, and Cybernetics*. Vol. 4, 1999, pp.430-434.

[21] Pedroso, J.P. Niche search: An application in vehicle routing. *IEEE World Congress on Computational Intelligence*, 1998, pp.177–182.

[22] Solomon, M.M. Algorithms for Vehicle Routing and Scheduling Problems with time window constraints. Northeastern University, Boston, Massachusetts, (December, 1985).

[23] Takeno, T., Tsujimura, Y. y Yamazaki, G. A single-phase method based on evolution calculation for vehicle routing problem. *4th Int. Conf. on Computational Intelligence and Multimedia Applications*, 2001, pp.103–07.

[24] Tan, K.C., Lee, T.H., Ou, K. y Lee, L.H. A messy genetic algorithm for the vehicle routing problem with time window constraints. *Congress on Evolutionary Computation*, Vol. 1, 2001, pp.679-686.

[25] Van Veldhuisen, D. A. Multiobjective Evolutionary Algorithms: Classifications, Analyses and New Innovations. PhD thesis, Department of Electrical and Computer Engineering. Graduate School of Engineering. Air Force Institute of Technology. Ohio, USA. (May, 1999).

[26] Yoshiike, N.y Takefuji, Y. Vehicle routing problem using clustering algorithm by maximum neural networks. *2nd Int. Conf. on Intelligent Processing and Manufacturing of Materials.* Vol. 2, 1999, pp.1109-1113.