PST-Prim Heuristic for the Open Vehicle Routing Problem

B. Campo, and A. Mendoza

***Abstract*—The objective of this paper is to present and demonstrate the validation of the functioning of a recursive heuristic based on the algorithm of Prim and that gives solution to the open vehicle routing problem (OVRP). Today this problem has a considerable approach, so a literature review that sets the theoretical basis for the work, is made. The OVRP is formally shown and the covering tree with paths (PST) is defined. Next, the subroutine that follows the PST-Prim algorithm is indicated to construct the PST of any graph, as well as the modification that must be made to arrive at the PST-Prim Heuristic that gives solution to the OVRP. An illustrative example of the construction of a PST to an example graph is presented. To illustrate and validate its effectiveness of the heuristic, it is used to solve 17 widely used problems to verify and compare the behavior of this type of algorithms. In addition, the PST-Prim heuristic performance is compared with other seven algorithms; the value of the objective function and the computation time for each algorithm on the 17 instances is presented. The PST-Prim propose the best solutions on 12 of the 17 instances. At the end, the performance, use and importance of the heuristic is discussed.**

***Index Terms*—Path Spanning Tree, Prim algorithm, PST-Prim heuristic, Open vehicle routing problem.**

I. Introducción

En problemas de enrutamiento de vehículos se pretende diseñar las rutas de una flota de transporte que sirve a un conjunto de clientes. Un tipo particular de estos problemas es el de enrutamiento abierto de vehículos u OVRP (*open vehicle routing problem*, por sus siglas en inglés) que es una variante trivial del clásico VRP (*vehicle routing problem*, por sus siglas en inglés), y difiere de él en que una vez los vehículos sirven a todos los nodos que les han sido asignados no tienen la obligación de volver al depósito (camino hamiltoneano). En otras palabras, cada ruta en el OVRP parte desde el depósito inicial, y termina necesariamente en un cliente. Desde esta perspectiva, el OVRP cuenta con un gran atractivo para organizaciones que optan por contratar, arrendar, o tercerizar una flota de vehículos en lugar de adquirir una propia. Por si fuera poco, un OVRP es más amigable con el medio ambiente que un VRP, ya que los vehículos no deben retornar al depósito principal lo que acorta el tiempo de servicio, y, si bien en principio parece más costos alquilar vehículos, los costos de mantenimiento de una flota propia no ocurren. Por esto, el OVRP se clasifica en la categoría de problemas de logística de terceros o 3PL (*third party logistics*, por sus siglas en inglés). [1].

B. Campo y A. Mendoza, are with the Engineering Faculty, Universidad del Atlántico, Barranquilla, Atlántico, Colombia, e-mail: bdcampo@uniatlantico.edu.co; adelmenoza@uniatlantico.edu.co.

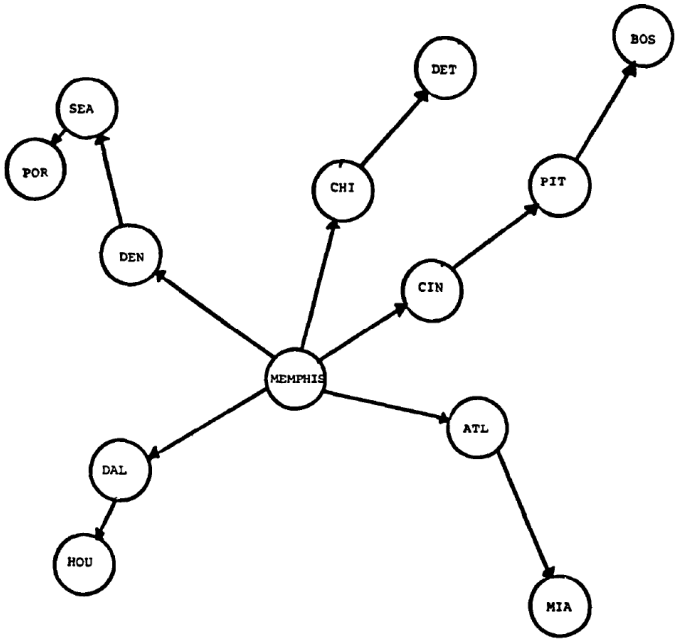


Fig. 1. Imagen original del primer esquema de un OVRP presentado en un trabajo de investigación científica. Tomado de [3].

A. Marco Histórico

El OVRP, de la mano de [2], se introdujo dos decenios después del clásico VRP. En esa primera oportunidad solo se propuso -de manera académica- el problema pero no se le dio nombre, ni formulación formal, ni propuesta de solución alguna. Poco más tarde, [3] modelaría por primera vez una situación real con este problema (ver Fig. 1), presentando entonces su primera propuesta de solución a través del algoritmo de los ahorros o de Clark and Wright. Seguirían algunas propuestas como las de [4], [5], [6] y [7], antes de que [8] realizaran su formulación formal y le otorgaran el nombre con el que hoy se conoce. La primera propuesta de modelo de programación entera binaria para el OVRP clásico, de acuerdo al conocimiento de los autores, se presenta en [7].

B. Importancia del OVRP

El interés de los autores en el OVRP es motivado por su importancia práctica y teórica. En un sentido práctico y desde una perspectiva empresarial, numerosas actividades de distribución en el mundo real encajan en el marco del OVRP. Por ejemplo, el reparto de periódicos, la generación de rutas de entrega con flota de aviones, el transporte ferroviario de mercancías, la distribución de productos lubricantes, el

transporte de material en una mina de carbón, el transporte escolar, la entrega de comidas escolares, la disposición de cables eléctricos en parques eólicos marinos, el sistema de transporte de un centro de distribución del canal retail, el transporte de pasajeros en buses intermunicipales , entre otros.

En un plano científico, la importancia del OVRP radica en que es un problema de optimización combinatoria del tipo NP-hard. Resolver el OVRP, esto es, encontrar una solución pseudóptima en un tiempo máximo de cómputo preestablecido, implica construir el mejor camino hamiltoneano para todos y cada uno de los vehículos. Como dicha asignación es de naturaleza combinatoria NP-Hard, el OVRP también lo es [9].

C. Estado del Arte

Para intentar dar solución al OVRP se han implementado distintas técnicas heurísticas y metaheurísticas en los últimos años. Las heurísticos construyen o buscan encontrar soluciones pseudóptimas dentro del conjunto de puntos del espacio muestral de manera aleatoria e iterativa. Una solución pseudóptima es la que, sin garantía de ser el óptimo global, mejora las soluciones actuales, normalmente empíricas, y a veces construidas por prueba y error en la vida real. Dentro de las que últimamente han sido usadas para dar solución al OVRP podemos resaltar por ejemplo: voraz o miope con base en el algoritmo de Kruskal [10], Clark and Wright [11].

La hiperheurísticas también han sido utilizadas. Destacan especialmente [12], [13].

Un papel preponderante, dentro del conjunto de técnicas implementadas para dar solución, no solo al OVRP, ha sido el de los métodos metaheurísicos (sobre aplicaciones de las metaheurísticas en otros VRP tipo, ver los trabajos de [14], [15] y [16]). Estos métodos, como los heurísticos, buscan de manera iterativa soluciones pseudóptimas, pero a diferencia de ellos, sus algoritmos están inspirados e construidos para imitar un patrón de comportamiento o ciertos fenómenos de la naturaleza. Dentro de los que se han implementado para tal fin se encuentra la búsqueda tabú (TS) [17], [18], la optimización por colonia de hormigas (ACO) [19], [20], el algoritmo de búsqueda gravitacional (GSA) [21], [22], el algoritmo de optimización por apareamiento de abejorros (BBMO) [23], el algoritmo del imperialista competitivo (ICA) [24], búsqueda local iterativa (ILS) [25], [26], algoritmos genéticos (GA) [27], algoritmo del virus de la influenza (PMS-MOIVA) [28], algoritmo del enjambre de partículas (PSO) [29], búsqueda amplia y adaptativa del vecindario (ALNS) [30], búsqueda de vecindad variable (VNS) y sus variantes [31], [32], [33], [34], entre otros.

Una revisión más extendida de la literatura y que abarca desde los primeros trabajos sobre el OVRP y sus métodos de solución se puede encontrar en [9].

En concreto, este trabajo presenta la heurística PST-Prim de naturaleza miope y voraz, que da solución al problema de enrutamiento abierto de vehículos. Primeramente se presenta formalmente el OVRP y su modelo de grafos. En seguida, se define el árbol recubridor con caminos. La sección siguiente muestra el algoritmo y se ilustra su funcionamiento con el grafo de un OVRP ejemplo. Hecho esto, se presenta la Heurística PST-Prim como una adaptación del algoritmo PST-Prim a la solución del problema de enrutamiento abierto de vehículos. Esta heurística se emplea para dar solución a 17 problemas comparativos de amplio uso en la validación de este tipo de métodos; estos resultados se comparan con otras siete heurísticas constructivas. Al final se hace la discusión sobre los resultados del trabajo y las sugerencias para trabajos futuros.

II. Descripción y Modelo del OVRP

El OVRP se define como un grafo no dirigido $G = (V, A)$, donde el nodo cero v_0 es el depósito y $N = \{v_1, \ldots, v_n\}$ es el conjunto de clientes. De esta manera, $V = \{v_0, \ldots, v_n\}$ es el conjunto de todos los nodos del grafo G, con $v_0 \cap N = \emptyset$. Todo nodo v_i en N $(1 \leq i \leq n)$ tiene una demanda asociada constante $q_i > 0$; el depósito o nodo cero tiene demanda nula $(q_0 = 0)$.

El conjunto de arcos o aristas se denota por A. Cada arco (i, j) en A, con $i \neq j$ y $j \neq 0$, tiene asociado una distancia constante $d_{ij} > 0$ que es la que un vehículo debe recorrer para ir del nodo i al j.

Un vehículo se considera activo si visita al menos un cliente.

El objetivo del problema es construir m rutas que permitan servir a n clientes, de tal manera que la operación total cueste lo menos posible, satisfaciendo las siguientes restricciones:

1) Cada ruta comienza en el depósito, y finaliza una vez se haya visitado el último de sus clientes asignados.
2) Cada cliente debe ser visitado una única vez por un único vehículo.
3) La demanda agregada de los nodos de cada ruta no debe exceder la capacidad Q de los vehículos.
4) La distancia recorrida de cada rutad no debe exceder la distancia recorrida máxima $L_{\text{máx}}$.

Un modelo de programación entera binaria del OVRP se puede construir con base en [35], [24], [33] como sigue: primero se definien un grupo de variables x_{ij}^k. Este modela la secuencia en la que los vehículos visitan y sirven a los clientes.

$$x_{ij}^k = \begin{cases} 1 & \text{si el nodo } j \text{ es visitado después del nodo } i \text{ por} \\ & \text{el vehículo } k; \\ 0 & \text{en otro caso.} \end{cases}$$

$$y_i^k = \begin{cases} 1 & \text{si el nodo } i \text{ es visitado por el vehículo } k; \\ 0 & \text{en otro caso.} \end{cases}$$

$$\min \left[\sum_{k=1}^{m} \sum_{i=0}^{n} \sum_{j=1}^{n} x_{ij}^k \right] \tag{1}$$

sujeto a:

$$\sum_{i=0}^{n} q_i y_i^k \leq Q, \ \forall k = 1, 2, \ldots, m; \tag{2}$$

$$\sum_{k=1}^{m} \sum_{i=0}^{n} x_{ij}^k = 1, \ \forall j = 1, 2, \ldots, n; \tag{3}$$

$$\sum_{k=1}^{m} \sum_{j=1}^{n} x_{ij}^k = 1, \ \forall i = 1, 2, \ldots, n; \tag{4}$$

$$\sum_{i=0}^{n} x_{iu}^{k} - \sum_{j=1}^{n} x_{uj}^{k} = 0,\ \forall i = 1, 2, \ldots, n, \quad \forall u = 1, 2, \ldots, n,\ \forall k = 1, 2, \ldots, m; \quad (5)$$

$$\sum_{(i,j) \in S \times S}^{n} x_{ij}^{k} \leq |S| - 1,\ \forall S \subseteq V : 1 \leq |S| \leq n, \quad \forall k = 1, 2, \ldots, m; \quad (6)$$

$$\sum_{j=1}^{n} x_{0j}^{k} = 1,\ \forall k = 1, 2, \ldots, m; \quad (7)$$

$$\sum_{i=1}^{n} x_{i0}^{k} = 0,\ \forall k = 1, 2, \ldots, m; \quad (8)$$

$$x_{ij}^{k} \in \{0, 1\},\ \forall k = 1, 2, \ldots, m,\ \forall i, j = 1, 2, \ldots, n; \quad (9)$$

$$y_{i}^{k} \in \{0, 1\},\ \forall k = 1, 2, \ldots, m,\ \forall i = 1, 2, \ldots, n; \quad (10)$$

La ecuación (1) es la función objetivo del OVRP, y modela la minimización de los costos de la operación. Las inecuaciones (2) garantizan que la demanda agregada de los nodos de cada ruta no exceda la capacidad de los vehículos. Las ecuaciones (3) y (4) aseguran que cada nodo es servido por un único vehículo. Por su parte, (5) es la típica restricción de flujo que garantiza la continuidad en los caminos hamiltoneanos. Las inecuaciones (6) describen la restricción de eliminación de subtours -o SEC por sus siglas en inglés *subtour elimination constraint*-. Ésta en particular es la introducida por [36] para el TSP, y asegura que no se obtengan subciclos (hamiltoneanos) indeseados en la solución (para SECs del tipo MTZ y otras, el lector interesado puede dirigirse a los trabajos [37] y [38]). Las ecuaciones (7) y (8) garantizan que todos los vehículos tengan como punto de partida el depósito y que ninguno finalice su recorrido en éste. Por último, (9) y (10) expresan la naturaleza binaria de x_{ij}^{k} y y_{i}^{k}, respectivamente.

III. Árbol Recubridor con Caminos

La heurística que aquí se propone para dar solución al OVRP es del tipo voraz o miope y se basa en el algoritmo de Prim [39], [40], [41]. Con este último es posible encontrar un árbol recubridor mínimo (o MST por sus siglas en inglés: *minimum spaning tree*) en cualesquiera grafo no dirigido y conexo. Dichos árboles resultan ser subgrafos del original y para su construcción los nodos que lo constituyen disponen de grado de libertad (g.l.) no restringido.

Como una solución al OVRP no cuenta con esta última posibilidad (la de nodos con g.l. mayores de 2), es necesario definir un árbol recubridor solución con características que se ajusten a las restricciones particulares de este tipo de problemas. Un ejemplo de este tipo de contrucciones es la que se presenta en [42]. Este trabajo hace uso del k-*degree centre tree* k-DCT de [43], que es un árbol que parte de un *degree-constrained minimum spanning tree* DCMST para dar solución al VRP. En lo que respecta a los grados de libertad de los nodos, el k-DCT cuenta con las siguientes restricciones:

1) El depósito o nodo cero tiene grado de libertad entre uno y $2m$ ($1 \leq g.l._{0} \leq 2m$).
2) Todos los nodos cliente tienen grado de libertad igual dos ($g.l._{i} = 2, \forall i \in N$).

Brandão propone como solución al OVRP hacer uso del k-DCT, y una vez construido eliminar todos los arcos $(i, 0)$ en A. Con este último paso las rutas cerradas del k-DCT se convierten en abiertas.

En este trabajo se presenta la construcción de un árbol recubridor a partir de las características propias del OVRP. Esto se logra adoptando una versión adaptada de la primera restricción en grados de libertad del k-DCT, y admitiendo una relajación de la segunda:

1) El depósito o nodo cero tiene grado de libertad entre uno y m ($1 \leq g.l._{0} \leq m$).
2) Los nodos cliente tienen grado de libertad entre uno y dos ($1 \leq g.l._{i} \leq 2, \forall i \in N$).

Con esta relajación, a todo algoritmo que se emplee para dar solución al OVRP se le permite establecer, en criterio propio, los nodos que serán fin de ruta. El árbol recubridor que resulta de admitir esta relajación lleva por nombre Árbol Recubridor con Caminos o PST (por sus siglas en inglés *path spanning tree*).

A. Un Ejemplo Gráfico

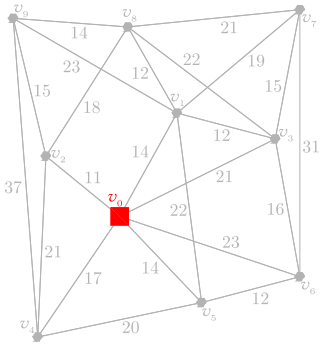
Considérese el grafo $G = (V, A)$ que se observa en la Fig. 2a. El nodo de inicio es v_{0} y se representa con un recuadrado rojo, los nodos cliente $N = \{v_{1}, \ldots, v_{9}\}$ se representan con puntos grises circulares, mientras que las aristas se representan con segmentos de recta del mismo color. El peso de éstas últimas es el número que las acompaña.

Las Fig. 2b y 2c son ambas soluciones propuestas por la heurística PST-Prim sobre el mismo problema. A cada una de ellas se llega ejecutando el algoritmo sucesivamente y rompiendo de manera diferente los empates que se presentan eventualmente en las iteraciones de cada corrida. La propuesta de la Fig. 2b tiene valor de la función objetivo 125, mientras que la de la Fig. 2c tiene valor de la función objetivo 124. Este fenómeno en el que se puede enmascarar una solución de mejor calidad por la aleatoriedad con la que se rompen los empate se evita ejecutando el algoritmo varias veces sobre el mismo problema. Note que hasta este punto y en este ejemplo no se ha tenido en cuenta restricción alguna propia del OVRP.

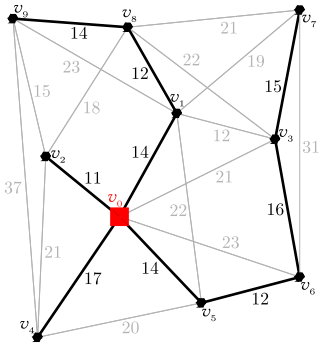
IV. Heurística PST-Prim

En el contexto de la heurística PST-Prim, dar solución al OVRP es encontrar un PST mínimo que satisfaga las restricciones de capacidad y recorrido máximo. Para ello, en este trabajo se presenta una modificación al algoritmo de Prim, que en su versión original comienza en cualquier vértice que se desee, y en cada iteración busca la arista que cuente con el menor peso de todas las que parten del MST actual y terminan en algún nodo no ingresado. Es así como el algoritmo de Prim evita la aparición de subciclos. Este proceso termina cuando todos los nodos del grafo original se han insertado en el MST.

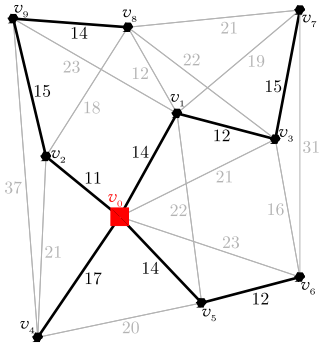
Contrario al algoritmo de Prim, la heurística PST-Prim requiere que se tenga siempre como nodo de partida v_{0}. La



(a) Modelo de grafos ejemplo.



(b) Solución 1. FO = 125.



(c) Solución 2. FO = 124.

Fig. 2. Grafo ejemplo y dos propuestas de MST distintas sobre el mismo.

heurística propuesta busca en cada iteración el arco con menor peso que parte o bien de un nodo ¡¡frente de rama¿¿ dentro del PST o del nodo cero, y termina en un nodo no ingresado del grafo original. Un nodo frente de rama es el que hasta la iteración actual ha sido ingresado al PST en construcción y cuenta con grado de libertad igual a uno ($g.l. = 1$). Como en el algoritmo de Prim, este procedimiento garantiza la ausencia de subciclos en los caminos hamiltoneanos.

Contextualizando al problema de enrutamiento abierto de vehículos, el PST es un árbol que parte desde el depósito (nodo cero), y en cada iteración va agregando el domicilio (nodo) no enrutado que está mas cerca, o bien al depósito o al último domicilio ingresado en cada ruta. Esta manera de interpretar el funcionamiento básico del algoritmo es más natural en tanto que parte de una aplicación real.

El Algoritmo 1 presenta el pseudocódigo que describe el funcionamiento de la heurística PST-Prim. Nótese en la subrutina 3 la introducción, de manera recursiva, de la restricción de capacidad y de recorrido máximo; eliminar esta línea significa obtener el algoritmo PST-Prim.

V. Resultados Computacionales

El algoritmo PST-Prim, así como los algoritmos seleccionados, se programaron en Julia Language. Se usó un computador con procesador Intel Pentium N3520 a 2.17 GHz, con 4 Gb de memoria RAM, y con Arch Linux como sistema operativo.

Se probó el algoritmo en una selección de instancias OVRP comparativas de uso recurrente; son ellas las propuestas por [44] (C), y [45] (F). Como característica general, todos los problemas son geométricamente simétricos, por lo que la distancia entre nodo y nodo es euclídea. Los grafos se generan a partir de *datasets* provistos por sus autores alojados en repositorios de acceso libre en internet. La Tabla I resume los parámetros requeridos para esta comparación.

Los algoritmos con los que se comparó el desempeño del algoritmo son todos constructivos: vecino más cercano (NNH) y vecino más cercano modificado (NNH mod) descritos en [9], de los ahorros (CW) descrito en [11] (con $\lambda = 2$ como se sugiere en el trabajo mencionado), de los pétalos (Sweep) usado en [21], de los pétalos aleatorio (Sweep Al) que es una propuesta de los autores en la que la fase II de cada ruta se ha aleatorizado. Se incluyó en la comparativa el algoritmo trivial y el algoritmo aleatorio que son de uso extendido en fases constructivas en metaheurísticas poblacionales.

A manera de ilustración, en la Fig. 3 se observa de manera gráfica las distintas propuestas de solución que los algoritmos seleccionados realizan sobre la instancia C1. El nodo cero o depósito se señala en las gráficas con un recuadro rojo. Los nodos cliente se ilustran con puntos negros; el tamaño del punto es una medida relativa de la demanda de cada nodo.

Para evaluar y comparar el desempeño de la heurística PST-Prim se usó el valor de la función objetivo (FO) y el tiempo de cómputo (CPU), como es costumbre en este tipo de comparativas. En la Tabla II se observan los valores de la función objetivo FO y el tiempo de cómputo CPU en milisegundos de las propuestas que cada algoritmo hizo sobre cada instancia. Aprovechando la eficiencia del lenguaje de programación empleado, cada algoritmo se ejecutó 1000 veces sobre cada instancia y se tomó la mejor propuesta arrojada, paliando el enmascaramiento de soluciones por los empates que pudieran presentarse. En la Tabla II se puede observar en negrillas el menor valor de FO y en cursiva el menor consumo de recursos computacionales por instancia (fila).

VI. Conclusiones

En este artículo se presentó la heurística PST-Prim, inspirada en el algoritmo de Prim, y concebida para dar solución

Algoritmo 1 Pseudocódigo de la heurística PST-Prim.

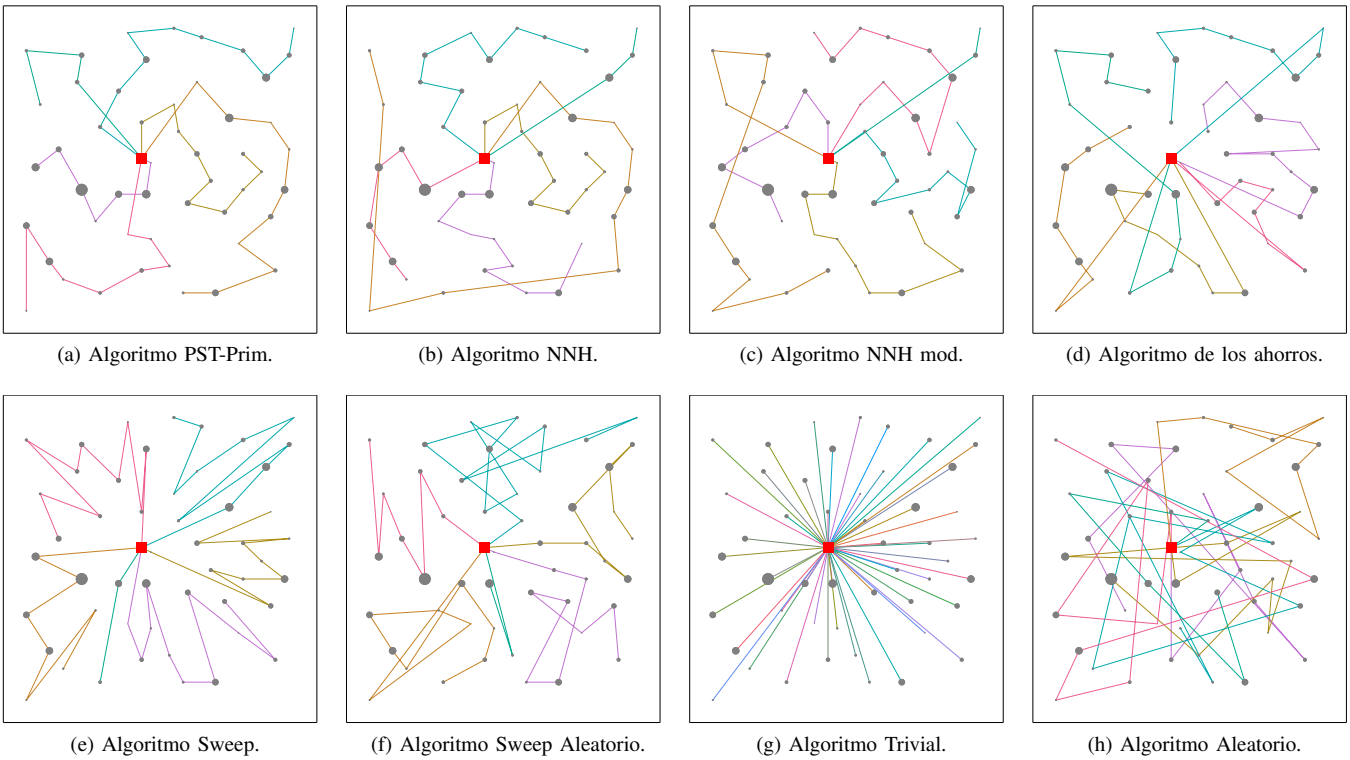
Entrada: $G = (V, A)$: Grafo del modelo; $d_{ij} \in \mathbf{D}$: La distancia (o peso) asociado a los arcos $(i, j) \in A$; $\mathbf{D}$: matriz de distancias de G; Q: capacidad de los vehículos; $L_{\text{máx}}$: distancia máxima de las rutas.

Salida: $\mathbf{R}$: arreglo de rutas del PST.

1: **while** Número de nodos en el PST actual $<$ Número total de nodos en G **do**
2: Buscar el nodo $j \in V, j \notin$ PST con la distancia d_{ij} mínima a los nodos $i \in$ PST$|g.l._i = 1$, o bien al nodo cero.
3: **if** q_j no produce un exceso sobre Q y d_{ij} no produce exceso sobre $L_{\text{máx}}$ **then**
4: Agregar v_j al PST y darle el nombre de nodo frente de rama ($g.l._j = 1$).
5: $g.l._i = 2$.
6: **end if**
7: **end while**

TABLA I
PARÁMETROS PRINCIPALES DE LOS CONJUNTOS DE INSTANCIAS SELECCIONADOS

Datasets	Christofides, Mindozzi y Toth [44]														Fisher [45]		
Instancias	**C1**	**C2**	**C3**	**C4**	**C5**	**C6**	**C7**	**C8**	**C9**	**C10**	**C11**	**C12**	**C12**	**C14**	**F1**	**F2**	**F3**
n	50	75	100	150	199	50	75	100	150	199	120	100	120	100	45	72	135
Q	160	140	200	200	200	150	140	200	200	200	200	200	200	200	2010	30000	2210
$L_{\text{máx}}$	∞	∞	∞	∞	∞	200	160	230	200	200	∞	∞	720	1040	∞	∞	∞



(a) Algoritmo PST-Prim. (b) Algoritmo NNH. (c) Algoritmo NNH mod. (d) Algoritmo de los ahorros.

(e) Algoritmo Sweep. (f) Algoritmo Sweep Aleatorio. (g) Algoritmo Trivial. (h) Algoritmo Aleatorio.

Fig. 3. La instancia 1 de [44] solucionada por los ocho algoritmos comparados. En cada solución propuesta, cada ruta se han graficado en un color diferente.

al problema de enrutamiento abierto de vehículos. Esta, a su vez, se basa en el algoritmo PST-Prim para construir el árbol recubridor con caminos de un grafo. Este último también definido en este trabajo. Además de describir su funcionamiento, se usó para dar solución al OVRP sobre 17 instancias de uso extendido. Su desempeño se comparó con otras ocho técnicas empleadas en la literatura, o bien para proveer una solución al OVRP, o como fases constructivas en algoritmos metaheurísticos.

Como se observa en la Fig. 2, el algoritmo PST-Prim es eficaz para construir un árbol recubridor con caminos, y su adaptación (la heurística PST-Prim) lo es para construir una solución que cumple con las restricciones propias del OVRP (ver Fig. 3). Además, no solo es útil para proponer soluciones a este tipo de problemas, sino para ser empleado como algoritmo de arranque de heurísticas de mejora o metaheurísticas.

El desempeño del algoritmo sobre el de los demás, en cuanto al alcance de menores valores de la función objetivo, se observa en la Tabla II. En ésta, la heurística de Prim propuso soluciones para todas las instancias y en 12 (de 17) de ellas fueron mejores que las propuestas de los demás algoritmos.

Si bien, el consumo de recursos computacionales por parte de la heurística propuesta no es la mejor, se debe tener en cuenta que los órdenes de magnitud en los que se desempeñan todos los algoritmos se encuentra en los milisegundos. Estas magnitudes en problemas de la vida real y en fases constructivas de algoritmos metaheurísticos (que se ejecutan una única vez al principio de toda la ejecución) son despreciables. Además, la eficiencia de los algoritmos dependen normalmente de la pericia de quien programa. Por su parte, se debe resaltar que el desempeño computacional alcanzado se ve favorecido al usar como lenguaje de programación Julia Language, que ha cobrado gran relevancia como lenguaje de análisis numérico de alto desempeño. Este lenguaje permitió la ejecusión repetitiva de los algoritmos, paliando al máximo el fenómeno de enmascaramiento de soluciones por empates.

El algoritmo más eficiente es el trivial, sin embargo es un algoritmo que solo se emplea en etapas de arranque de algoritmos metaheurísticos, ya que carece de sentido práctico en una eventual aplicación de la vida real, a diferencia del

TABLA II
DESEMPEÑO DE CADA ALGORITMO SOBRE CADA INSTANCIA (CPU EN [MS])

	Prim		NNH		NNH mod		CW		Sweep		Sweep Al		Trivial		Aleatorio	
	FO	CPU	FO	CPU	FO	CPU	FO	CPU	FO	CPU	FO	CPU	FO	CPU	FO	CPU
C1	**443,39**	8,32	519,32	0,37	511,83	0,37	629,08	99,48	704,78	0,36	722,45	0,49	1201,17	*0,03*	1247,36	0,15
C2	**683,61**	21,62	723,12	0,84	688,51	0,85	951,98	335,06	1068,08	0,57	972,39	0,84	1836,73	*0,04*	2049,81	0,31
C3	**702,30**	26,23	795,04	1,42	747,08	1,32	1021,44	888,39	1204,29	0,70	1348,32	0,89	2512,96	*0,05*	2941,99	0,26
C4	947,66	62,52	**895,45**	3,08	920,03	2,86	1256,92	4339,58	1812,11	1,05	1777,32	1,38	3698,50	*0,07*	4368,19	0,47
C5	**1009,35**	117,16	1021,10	5,61	1084,77	5,21	1428,08	13272,25	2214,19	1,37	2264,83	1,80	4822,45	*0,10*	5857,81	0,89
C6	**464,67**	8,41	488,01	0,39	513,02	0,38	641,23	94,65	702,43	0,38	694,71	0,50	1201,17	*0,02*	1174,25	0,34
C7	**683,61**	22,16	717,57	0,87	684,79	0,81	951,98	333,52	1068,08	0,54	957,77	0,80	1836,73	*0,04*	1968,60	0,80
C8	**702,30**	26,17	784,13	1,67	747,08	1,71	1021,44	899,97	1204,29	0,69	1288,28	60,80	2512,96	*0,04*	2809,86	0,91
C9	947,66	72,76	**895,45**	3,74	920,03	3,76	1256,92	4298,52	1824,31	1,06	1755,33	6,28	3698,50	*0,06*	4228,58	1,09
C10	**1009,83**	131,57	1034,18	6,82	1084,77	6,07	1428,08	13158,15	2234,17	1,37	2195,09	3,45	4822,45	*0,08*	5665,04	1,25
C11	832,09	30,71	**769,72**	1,94	831,04	1,95	1084,15	1801,80	2674,10	0,85	3291,93	1,04	6128,53	*0,05*	5470,77	0,26
C12	**601,37**	33,27	653,23	1,76	734,69	1,73	741,18	884,46	856,60	0,71	1166,00	0,95	2909,15	*0,04*	3388,66	0,25
C13	832,09	32,54	**768,36**	2,04	831,04	1,83	1084,15	1806,59	2674,10	0,83	2956,89	2,18	6128,53	*0,05*	5368,55	0,59
C14	**601,37**	24,70	649,02	1,80	734,69	1,92	741,18	887,92	856,60	0,71	1224,77	0,95	2909,15	*0,04*	3420,02	0,22
F1	**573,25**	5,88	643,15	0,28	613,30	0,31	792,95	55,33	993,03	0,42	1082,20	0,52	1791,35	*0,02*	1827,74	0,10
F2	**191,18**	8,29	228,11	0,69	260,05	0,71	264,98	259,94	357,58	0,57	539,39	0,68	1084,66	*0,05*	1022,29	0,40
F3	942,97	26,93	**924,13**	2,40	935,62	2,46	1345,06	2933,80	1991,49	1,07	2455,99	1,32	4884,05	*0,08*	5421,92	0,84

algoritmo PST-Prim que intenta abarcar también esto último.

Para trabajos posteriores, se sugiere emplear el PST-Prim como heurística constructiva en fases de arranque de técnicas metaheurísticas, así como un estudio de su complejidad computacional. Se sugiere además realizar la adaptación del mismo para dar solución a problemas OVRP multi depósito. Se sugiere la construcción de una heurística similar inspirada en el algoritmo de Borůvca, que como el algoritmo de Prim, se usa para construir un MST sobre un grafo.

AGRADECIMIENTOS

El primer autor agradece el apoyo desinteresado del Ing. Hernando Andrés Rueda Bustillo. S.D.G.

REFERENCIAS

[1] A. R. Hedar and M. Abdallah, "Applying tabu search in finding an efficient solution for the ovrp," *International Journal of Open Problems in Computer Science and Mathematics*, vol. 7, no. 4, pp. 36–51, 12 2014.

[2] L. Schrage, "Formulation and structure of more complex/realistic routing and scheduling problems," *Networks*, vol. 11, no. 2, pp. 229–232, 1981.

[3] S. Raff, "Routing and scheduling of vehicles and crews: The state of the art," *Computers & Operations Research*, vol. 10, no. 2, pp. 6 369 117 149 195–67 115 147 193 211, 1983.

[4] J. Ben Atkinson, "A vehicle-scheduling system for delivering school meals," *Journal of the Operational Research Society*, vol. 41, no. 8, pp. 703–711, Aug. 1990. [Online]. Available: http://www.jstor.org/stable/2583475

[5] Z. Fu and M. Wright, "Train plan model for british rail freight services through the channel tunnel," *Journal of the Operational Research Society*, vol. 45, no. 4, pp. 384–391, 1994.

[6] C. R. D. Serna and J. P. Bonrostro, "Minmax vehicle routing problems: application to school transport in the province of burgos," in *Computer-Aided Scheduling of Public Transport*. Springer, 2001, pp. 297–317.

[7] L. Li and Z. Fu, "The school bus routing problem: a case study," *Journal of the Operational Research Society*, pp. 552–558, 2002.

[8] D. Sariklis and S. Powell, "A heuristic method for the open vehicle routing problem," *Journal of the Operational Research Society*, pp. 564–573, 2000.

[9] B. Campo-Zúñiga and A. Mendoza, "Propuesta de un modelo de ruteo de vehículos abierto en una institución prestadora de servicios de salud," *Entramado*, vol. 14, no. 2, pp. 288–298, jun 2018. [Online]. Available: https://doi.org/10.18041/1900-3803/entramado.2.4761

[10] E. Ruiz, O. Gómez, J. Reyes, P. Sarabia, and E. Villegas, "Greedy algorithm for the open vehicle routing problem," in *Memorias del séptimo Congreso Internacional de Ingeniería Arquitectura y Diseño VÉRTICE 2014*, J. Nieto, J. Aguilar, and L. Lim, Eds. Universidad Autónoma de Baja California, Apr. 2014, pp. 22–26.

[11] T. Pichpibul and R. Kawtummachai, "A heuristic approach based on clarke-wright algorithm for open vehicle routing problem," *The Scientific World Journal*, vol. 2013, 2013.

[12] R. Tyasnurita, E. Özcan, and R. John, "Learning heuristic selection using a time delay neural network for open vehicle routing," in *Evolutionary Computation (CEC), 2017 IEEE Congress on*. IEEE, 2017, pp. 1474–1481.

[13] L. Sun and B. Wang, "A goal-robust-optimization approach for solving open vehicle routing problems with demand uncertainty," *Wireless Personal Communications*, vol. 103, no. 1, pp. 1059–1075, feb 2018. [Online]. Available: https://doi.org/10.1007/s11277-018-5496-9

[14] P. N. M. Dorantes, P. H. I. Sanchez, J. M. V. Cantu, E. L. Garcia, and G. M. Mendez, "Design and optimization of distribution routes using evolutionary strategy and type-1 singleton neuro-fuzzy systems," *IEEE Latin America Transactions*, vol. 16, no. 5, pp. 1499–1507, 2018.

[15] C. Lagos, G. Guerrero, E. Cabrera, A. MOLTEDO-PERFETTI, F. Johnson, and F. Paredes, "An improved particle swarm optimization algorithm for the vrp with simultaneous pickup and delivery and time windows," *IEEE Latin America Transactions*, vol. 16, no. 6, pp. 1732–1740, 2018.

[16] B. de Athayde Prata, "A hybrid genetic algorithm for the vehicle and crew scheduling in mass transit systems," *IEEE Latin America Transactions*, vol. 13, no. 9, pp. 3020–3025, 2015.

[17] Y. Niu, Z. Yang, P. Chen, and J. Xiao, "A hybrid tabu search algorithm for a real-world open vehicle routing problem involving fuel consumption constraints," *Complexity*, vol. 2018, pp. 1–12, 2018. [Online]. Available: https://doi.org/10.1155/2018/5754908

[18] ——, "Optimizing the green open vehicle routing problem with time windows by minimizing comprehensive routing cost," *Journal of Cleaner Production*, vol. 171, pp. 962–971, jan 2018. [Online]. Available: https://doi.org/10.1016/j.jclepro.2017.10.001

[19] F. Maleki and M. a. Yousefikhoshbakht, "A hybrid algorithm for the open vehicle routing problem," *International Journal of Optimization in Civil Engineering*, vol. 9, no. 2, 2019. [Online]. Available: http://ijoce.iust.ac.ir/article-1-395-en.html

[20] M. YousefiKhoshbakht and N. M. Darani, "A combined metaheuristic algorithm for the vehicle routing problem and its open version," *Journal of AI and Data Mining*, vol. 7, no. 1, Mar 2019. [Online]. Available: http://doi.org/10.22044/jadm.2018.7116.1840

[21] A. A. R. Hosseinabadi, M. Kardgar, M. Shojafar, S. Shamshirband, and A. Abraham, "Gravitational search algorithm to solve open vehicle routing problem," in *Innovations in Bio-Inspired Computing and Applications*. Springer, 2016, pp. 93–103.

[22] A. A. R. Hosseinabadi, J. Vahidi, V. E. Balas, and S. S. Mirkamali,

"OVRP_GELS: solving open vehicle routing problem using the gravitational emulation local search algorithm," *Neural Computing and Applications*, vol. 29, no. 10, pp. 955–968, sep 2016. [Online]. Available: https://doi.org/10.1007/s00521-016-2608-x

[23] Y. Marinakis and M. Marinaki, "A bumble bees mating optimization algorithm for the open vehicle routing problem," *Swarm and Evolutionary Computation*, vol. 15, pp. 80–94, Apr. 2014.

[24] S. Shamshirband, M. Shojafar, A. A. R. Hosseinabadi, and A. Abraham, "Ovrp_ica: An imperialist-based optimization algorithm for the open vehicle routing problem," in *International Conference on Hybrid Artificial Intelligence Systems*. Springer, 2015, pp. 221–233.

[25] J. Brandão, "Iterated local search algorithm with ejection chains for the open vehicle routing problem with time windows," *Computers & Industrial Engineering*, vol. 120, pp. 146–159, jun 2018. [Online]. Available: https://doi.org/10.1016/j.cie.2018.04.032

[26] R. Atefi, M. Salari, L. C. Coelho, and J. Renaud, "The open vehicle routing problem with decoupling points," *European Journal of Operational Research*, vol. 265, no. 1, pp. 316–327, feb 2018. [Online]. Available: https://doi.org/10.1016/j.ejor.2017.07.033

[27] J. Dutta, P. S. Barma, S. Kar, and T. De, "A modified kruskal's algorithm to improve genetic search for open vehicle routing problem," *International Journal of Business Analytics*, vol. 6, no. 1, pp. 55–76, jan 2019. [Online]. Available: https://doi.org/10.4018/ijban.2019010104

[28] I.-D. Psychas, E. Delimpasi, M. Marinaki, and Y. Marinakis, "Influenza virus algorithm for multiobjective energy reduction open vehicle routing problem," in *Emergence, Complexity and Computation*. Springer International Publishing, 2018, pp. 145–161. [Online]. Available: https://doi.org/10.1007/978-3-319-77510-4_5

[29] L. Shen, F. Tao, and S. Wang, "Multi-depot open vehicle routing problem with time windows based on carbon trading," *International Journal of Environmental Research and Public Health*, vol. 15, no. 9, p. 2025, sep 2018. [Online]. Available: https://doi.org/10.3390/ijerph15092025

[30] R. Lahyani, A.-L. Gouguenheim, and L. C. Coelho, "A hybrid adaptive large neighbourhood search for multi-depot open vehicle routing problems," *International Journal of Production Research*, pp. 1–14, mar 2019. [Online]. Available: https://doi.org/10.1080/00207543.2019.1572929

[31] A. Subramanian, E. Uchoa, and L. S. Ochi, "A hybrid algorithm for a class of vehicle routing problems," *Computers & Operations Research*, vol. 40, no. 10, pp. 2519–2531, 10 2013.

[32] A. Reinholz and H. Schneider, "A hybrid (1+1)-evolutionary strategy for the open vehicle routing problem," in *Advances in Metaheuristics*. Springer, 2013, pp. 127–141.

[33] A. Z. Şevkli and B. Güler, "A multi-phase oscillated variable neighbourhood search algorithm for a real-world open vehicle routing problem," *Applied Soft Computing*, vol. 58, pp. 128–144, 2017.

[34] S. Kritzinger, K. F. Doerner, F. Tricoire, and R. F. Hartl, "Adaptive search techniques for problems in vehicle routing, part ii: A numerical comparison," *Yugoslav Journal of Operations Research*, vol. 25, no. 1, 2016.

[35] S. MirHassani and N. Abolghasemi, "A particle swarm optimization algorithm for open vehicle routing problem," *Expert Systems with Applications*, vol. 38, no. 9, pp. 11 547–11 551, 2011.

[36] G. Dantzig, R. Fulkerson, and S. Johnson, "Solution of a large-scale traveling-salesman problem," *Journal of the operations research society of America*, vol. 2, no. 4, pp. 393–410, 1954.

[37] T. Bektaş and L. Gouveia, "*Requiem* for the Miller-Tucker-Zemlin subtour elimination constraints?" *European Journal of Operational Research*, vol. 236, no. 3, pp. 820–832, Aug. 2014.

[38] U. Pferschy and R. Staněk, "Generating subtour elimination constraints for the TSP from pure integer solutions," *Central European Journal of Operations Research*, vol. 25, no. 1, pp. 231–260, Mar. 2017.

[39] V. Jarník, "O jistém problému minimálním. Z dopisu panu O. Borůvkovi," *Pracé Moravské Přírodovědecké Společnosti*, vol. 4, no. 4, pp. 57–63, 1930. [Online]. Available: https://dml.cz/handle/10338.dmlcz/500725

[40] R. C. Prim, "Shortest connection networks and some generalizations," *Bell Labs Technical Journal*, vol. 36, no. 6, pp. 1389–1401, 1957.

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

[42] J. Brandão, "A tabu search algorithm for the open vehicle routing problem," *European Journal of Operational Research*, vol. 157, no. 3, pp. 552–564, 2004.

[43] N. Christofides, A. Mingozzi, and P. Toth, "Exact algorithms for the vehicle routing problem, based on spanning tree and shortest path relaxations," *Mathematical programming*, vol. 20, no. 1, pp. 255–282, 1981.

[44] N. Christofides, A. Mingozzi, P. Toth, and C. Sandi, "Loading problems," in *Combinatorial Optimization*, N. Christofides, A. Mingozzi, and P. Toth, Eds. John Wiley & Sons Ltd, 1979, ch. 12, pp. 339–369, based on lectures held in SOGESTA, Urbino, 30th May-11th June 1977. [Online]. Available: https://lib.ugent.be/catalog/rug01:000046302

[45] M. L. Fisher, "Optimal solution of vehicle routing problems using minimum k-trees," *Operations research*, vol. 42, no. 4, pp. 626–642, 1994.