Solución para el problema de ruteo de vehículos con ventanas de tiempo utilizando cross docking (VRPTWCD) mediante la metaheurística búsqueda tabú

Ronaldo Garzón Peña

Camila Andrea Cantillo Quintero

Trabajo de Grado para optar por el título de Ingeniero Industrial

Director:

Carlos Eduardo Díaz Bohórquez

M.Sc. Ingeniería Industrial

Universidad Industrial de Santander

Facultad de Ingenierías Fisicomecánicas

Escuela de Estudios Industriales y Empresariales

Bucaramanga

2021

Agradecimientos

A Dios por darme salud, sabiduría y la oportunidad de cursar mis estudios profesionales.

A mis profesores y la Universidad Industrial de Santander por brindarme las herramientas y recursos necesarios para culminar con éxito esta etapa de mi vida.

A mi compañera de tesis *Camila Andrea Cantillo Quintero* por su colaboración y acompañamiento en esta etapa universitaria.

A nuestro director de proyecto *Carlos Eduardo Diaz Bohórquez* por su guía y disposición para la elaboración de este trabajo.

A mis padres *Wber Orlando Garzón Murillo* y *Jakeline Peña Arias* por sus enseñanzas a lo largo de mi vida y estar siempre para mí.

A mi hermano *Nicolas Garzón Peña* por su constante apoyo y ejemplo a seguir.

A mis compañeros que hicieron parte de este proceso.

Ronaldo Garzón Peña.

A mis padres por el amor, paciencia y por siempre estar motivándome a salir adelante.

A mi hermano por su apoyo y amor incondicional.

A mi compañero de proyecto Ronaldo Garzón Peña por su paciencia y colaboración.

A nuestro director de proyecto Carlos Eduardo Diaz Bohórquez por su acompañamiento y disposición en el desarrollo del proyecto.

A todos mis compañeros que hicieron parte de este proceso.

A mis profesores y a la Universidad Industrial de Santander por permitir la culminación de esta meta.

Camila Andrea Cantillo quintero.

Tabla de contenido

Lista de figuras

Lista de tablas

Lista de apéndices

Ver apéndices adjuntos y pueden ser consultados en la base de datos de la Biblioteca UIS

Resumen

Título: Solución para el problema de ruteo de vehículos con ventanas de tiempo utilizando cross docking (VRPTWCD) mediante la metaheurística búsqueda tabú*

Autores: Ronaldo Garzon Peña
Camila Andrea Cantillo Quintero**

Palabras clave: Ruteo de vehículos (VRP), Cross dock, ventanas de tiempo, algoritmo búsqueda tabú, cadena de suministro, Matlab, optimización.

Descripción:

En la presente investigación se aborda el problema de ruteo de vehículos con ventanas de tiempo y cross-dock (VRPTWCD), en donde se tienen dos flotas de vehículos completamente distintas, la primera debe atender a un conjunto de proveedores bajo unas restricciones de capacidad para cada vehículo, mientras que la segunda atiende un grupo de clientes con su respectiva demanda, los cuales presentan adicionalmente ventanas de tiempo en las que estos deben ser atendidos, todo dentro de una jornada laboral establecida.

El objetivo de esta investigación consiste en diseñar un algoritmo metaheurístico de búsqueda tabú, el cual parte de una solución inicial formulada a través de una heurística adaptada del vecino más cercano; posteriormente se programan y ejecutan en el software Matlab ® en su versión 2020b. Luego, se realiza un diseño experimental 2^3 con tres replicas para determinar los factores que tienen mayor influencia en la variable de respuesta

Finalmente, se evalúa el algoritmo con instancias adaptadas de la literatura y se comparan los resultados obtenidos para conocer que tanto mejora la solución obtenida de la búsqueda tabú con respecto a la solución inicial, teniendo en cuenta la calidad de la respuesta y los tiempos de cómputo, luego se presentan las conclusiones de la investigación y una serie de recomendaciones para futuros trabajos de esta temática.

---

* Proyecto de grado
** Facultad de ingeniería Físico Mecánicas. Escuela de Estudios Industriales y Empresariales. Programa de Ingeniería Industrial. Director M.Sc. Carlos Eduardo Díaz Bohórquez

Abstract

Title: Solution for the vehicle routing problem with time windows and cross docking (VRPTWCD) using a tabu search metaheuristic*

Authors: Ronaldo Garzon Peña
Camila Andrea Cantillo Quintero**

Keywors: Vehicle routing problem (VRP), Cross dock, time windows, Tabu search algorithm, supply chain, Matlab, optimization.

Description:

This research paper addresses the vehicle routing problem with time windows and a cross-dock where there are two vehicles fleets, the first must attend a set of suppliers under capacity restrictions, while the second one must attend a bunch of customers with their respective demand and they additionally have time windows which must be attended to, All of this should be done in a previously established working day.

The objective of this paper is to design a tabu search algorithm, which starts from an initial solution formulated through an adapted heuristic from the nearest neighbor, subsequently, they are programmed and executed in the Matlab ® software version 2020b. Later a design of experiment 2^3 is carried out with three replicas to find out the factors that have the most influence on the variable answer,

Finally, the algorithm is evaluated with adapted instances from the literature and the results are compared to find out how much the solution obtained from the tabu search improves concerning the initial solution, bear in mind the quality of the answer and the computational times, then the conclusions of the paper and some recommendations for future works on this topic are presented.

---

* Proyecto de grado

** Facultad de ingeniería Físico Mecánicas. Escuela de Estudios Industriales y Empresariales. Programa de Ingeniería Industrial. Director M.Sc. Carlos Eduardo Díaz Bohórquez

Introducción

La gestión de la cadena de suministro (SCM por sus siglas en inglés) es una actividad que toma cada vez más importancia a nivel empresarial, pues un óptimo manejo de ésta permite que las empresas puedan competir más fácilmente en un entorno de mercado cada vez más dinámico y globalizado. La distribución al ser la última etapa dentro de la cadena de suministro toma una gran importancia a nivel logístico dentro de la organización, pues en ella, no solo se ve reflejada la capacidad de cumplimiento de una empresa al hacer entrega de los productos en función del tiempo, calidad y precio; sino que también representa un gran impacto en los costos totales del proceso distributivo, los cuales se ven reflejados en el producto final, motivo por el cual resulta de vital importancia la reducción de éstos para dicha etapa.

Las medidas más utilizadas hoy en día para agilizar la etapa de distribución en la cadena de suministro son: Eliminación de paradas intermedias u ociosas, reducción de tiempos de espera al llegar a cada destino, utilización intensiva del cross-docking y reducción en el consumo de combustible mediante la optimización de las rutas (Caro, 2017). Esta última es de gran importancia para el proceso de gestión para la distribución, pues la asignación de rutas óptimas no solo permite la reducción de gastos en combustible, sino que también deriva en una reducción general de los tiempos de recogida y envío, incluyendo la posibilidad de atender un mayor número de clientes dependiendo de las capacidades de la empresa.

El cross-docking es una estrategia que ha ido tomando más importancia en los últimos años en la etapa distributiva para la gestión de la cadena de suministro, pues esta tiene como prioridad reducir los costos y tiempos de almacenaje, a la par de agilizar los tiempos de embarque para la pronta distribución de los productos, resultando en una herramienta muy importante cuando se trata de productos que van perdiendo su calidad con el paso del tiempo, como lo viene a ser el caso de productos perecederos en los que termina siendo más costoso la

pérdida del producto por una lenta gestión en la cadena de suministro al no implementar estrategias como esta, ya que el dinero invertido en las etapas anteriores termina siendo irrecuperable, de esta manera el cross-dock también actúa como un seguro para evitar la pérdida de productos al momento de realizar la distribución.

En el presente proyecto se llevará a cabo un planteamiento teórico para el problema de ruteo de vehículos con ventanas de tiempo y cross-dock (VRPTWCD por sus siglas en inglés) en donde se tienen dos flotas de vehículos, la primera recogerá la mercancía en las instalaciones de cada proveedor para posteriormente llevarla al centro de cross-dock, donde será clasificada y reembarcada en una segunda flota la cual se encargará de llevar los productos a los diferentes minoristas en unas ventanas de tiempo definidas, creando una situación que se asemeja a los problemas de la vida real, ya sea por dar cumplimiento a los requerimientos en las ventanas de tiempo establecidas por los clientes o por el lado de la logística, donde se busca disminuir gastos de almacenamiento y despachar rápidamente los pedidos gracias al centro de cross-dock, finalmente será diseñado un Algoritmo de búsqueda tabú (TS por sus siglas en inglés) con el fin de dar respuesta al problema en un tiempo computacional razonable.

Tabla 1

Cumplimiento de objetivos

Objetivo	Cumplimiento
Realizar una revisión de literatura sobre los artículos relacionados al problema de ruteo de vehículos y sus extensiones para las metodologías de ventanas de tiempo y cross-docking.	Capítulo 2, apéndices A, B.
Formular el modelo matemático para el VRPTWCD.	Capítulo 4.
Construir un algoritmo de búsqueda tabú para la solución del VRPTWCD.	Capítulo 5, apéndices G, H.

Continuación Tabla 1

Cumplimiento de objetivos

Verificar la eficiencia del algoritmo mediante instancias adaptadas de la literatura.	Capítulo 6, apéndices E, F, I.
Elaborar un artículo publicable a partir de la investigación realizada.	Apéndice J.

1. Generalidades del proyecto

1.1. Planteamiento del problema

En la actualidad las empresas luchan constantemente por destacarse sobre su competencia, la adecuada gestión de la cadena de suministro es una de las principales maneras para lograr este objetivo, pues ella se centra en agilizar el proceso de adquisición de materias primas, fabricación y distribución al consumidor final dependiendo del tipo de empresa. Desde el punto de vista de la logística, la asignación de rutas es una de las herramientas más importantes para lograr esta gestión, ya que su correcto manejo, permite obtener una reducción significativa en los costos transporte, entre los cuales se comprenden los siguientes: costos de embarque, costos de retrasos, costos de pérdidas, etc. adicionalmente la correcta asignación de rutas permite que los productos estén en manos del cliente en el momento que lo requiera (Lozada et al., 2012).

El problema de ruteo de vehículos (VRP) consiste en la correcta asignación de una flota la cual pueda satisfacer una determinada cantidad de clientes, de manera tal que se optimicen las rutas recorridas, en el VRP usualmente se tiene en cuenta la cantidad y capacidad de los vehículos, los lugares de destino y la demanda de los clientes (Rocha, 2011).

En la realidad algunos clientes no tienen disponibilidad para recibir los pedidos en cualquier momento del día, por eso este problema se hace mucho más complejo de lo que

aparenta, ya que hay que tener en cuenta los horarios de recepción para cada cliente, a esto último se le denomina ventana de tiempo, la cual es una ampliación del VRP original conocida como problema de ruteo de vehículos con ventanas de tiempo (VRPTW) (Montes et al., 2020). Sin embargo, para esta situación el incumplimiento de estas últimas puede ocasionar desde tiempos de espera, hasta costos extra por retrasos o adelantos a los tiempos pactados y en el peor de los casos cancelar la recepción del producto por parte del cliente.

En la gestión de la cadena de suministro también es importante disminuir los costos de distribución, pues estos pueden representar entre un 35% al 40% de los costos de transporte totales (González et al., 2010). Como complemento a lo dicho anteriormente, para disminuir los costos en esta etapa se recomienda la implementación de centros de cross docking, ya que estos tienen por objetivo hacer los tiempos de almacenamiento casi nulos ayudando a acortar los tiempos de espera de los clientes, logrando que esta herramienta se convierta en un factor fundamental para la solución de este problema (González & Becerra, 2017).

Como se pudo ver en lo mencionado anteriormente, existen diversas formas de abordar el VRP, implementando bien sea el cross docking, las ventanas de tiempo, el tipo de flota vehicular, entre otros. Con base en lo anterior y dependiendo de los objetivos del autor, es posible de apreciar distintas alternativas de solución propuestas para el VRP, los cuales varían según lo que se quiera estudiar, consideraciones del problema y el algoritmo a evaluar (Morais et al., 2014). Teniendo en cuenta lo anterior, se abordará un problema de ruteo de vehículos con ventanas de tiempo y cross-dock (VRPTWCD), el cual se resolverá por medio de un algoritmo metaheurístico conocido como búsqueda tabú, en donde se busca dar solución al problema y evaluar la eficiencia del algoritmo.

1.2. Objetivos

1.2.1 Objetivo general

Desarrollar un algoritmo metaheurístico de búsqueda tabú para el problema de ruteo de vehículos con ventanas de tiempo utilizando cross-docking.

1.2.2 Objetivos específicos

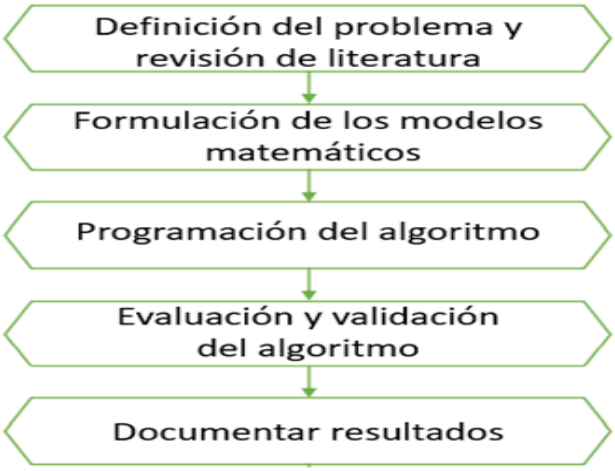
1. Realizar una revisión de literatura sobre los artículos relacionados al problema de ruteo de vehículos y sus extensiones para las metodologías de ventanas de tiempo y cross-docking.
2. Formular el modelo matemático para el VRPTWCD.
3. Construir un algoritmo de búsqueda tabú para la solución del VRPTWCD
4. Verificar la eficiencia del algoritmo mediante instancias adaptadas de la literatura.
5. Elaborar un artículo publicable a partir de la investigación realizada.

1.3. Metodología

A continuación, se muestra la estructura que se llevó a cabo para la ejecución de esta investigación, en donde se identifican las fases que permiten dar cumplimiento a los diferentes objetivos planteados en este proyecto, la cual se puede observar en la *figura 1*.

Figura 1.

Metodología



1.3.1. Etapa 1: Definición del problema y revisión de la literatura (Dando cumplimiento al objetivo 1).

- Definir las palabras clave idóneas que permitan realizar una búsqueda precisa en bases de datos científicas.
- Construir la ecuación de búsqueda
- Hacer uso de la base de datos Web Of Science suministrada por la Universidad Industrial de Santander, haciendo uso de la ecuación de búsqueda planteada previamente y depurando los resultados para obtener la información deseada.
- Analizar la información obtenida por medio de un análisis bibliométrico haciendo uso del software VOSviewer de acceso abierto.
- Realizar una revisión de la literatura obtenida sobre el problema de ruteo de vehículos con ventanas de tiempo y cross-dock junto a sus métodos de solución.

1.3.2. Etapa 2: Descripción del problema y formulación del modelo matemático (Dando cumplimiento al objetivo 2).

- Comprender la estructura básica de los modelos matemáticos relacionados con el ruteo de vehículos y sus variantes.
- Definir la estructura del modelo con base en las diferentes variantes encontradas durante la fase investigativa de la literatura y las condiciones que especifique el investigador.
- Instaurar claramente el objetivo del modelo.
- Definir los parámetros del modelo.
- Determinar las variables de solución del modelo.
- Precisar las restricciones que se deben cumplir en el modelo matemático.

La formulación del modelo matemático se llevará a cabo a partir de dos fases:

- la primera parte consiste en un problema de ruteo de vehículos capacitados (vehículos con capacidad limitada), los cuales se encargarán de atender exclusivamente a los proveedores y llevar la mercancía recogida al centro de cross-dock.
- La segunda parte es un problema de ruteo de vehículos capacitados, los cuales se encargarán de atender exclusivamente a los minoristas, quienes tendrán sus respectivas ventanas de tiempo y dentro de una hora límite de operación.

Una vez unificados los dos modelos anteriores, estos permiten dar solución al problema de VRPTWCD.

La representación gráfica de las partes 1 y 2 se pueden observar en las figuras 2 y 3 respectivamente.

Figura 2.

Representación gráfica del modelo para la parte 1.

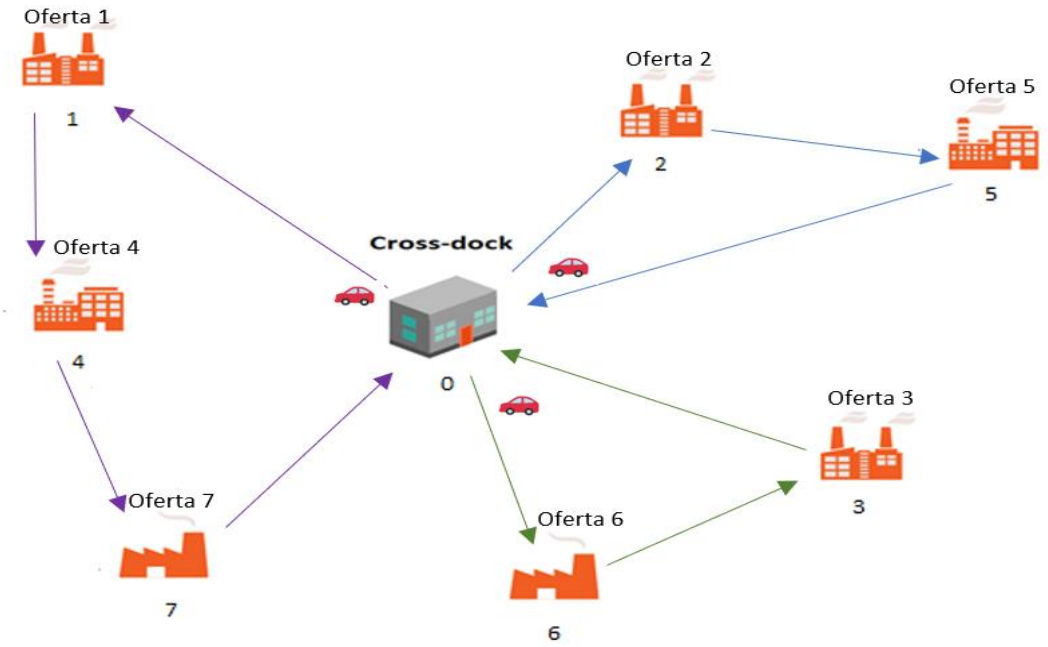
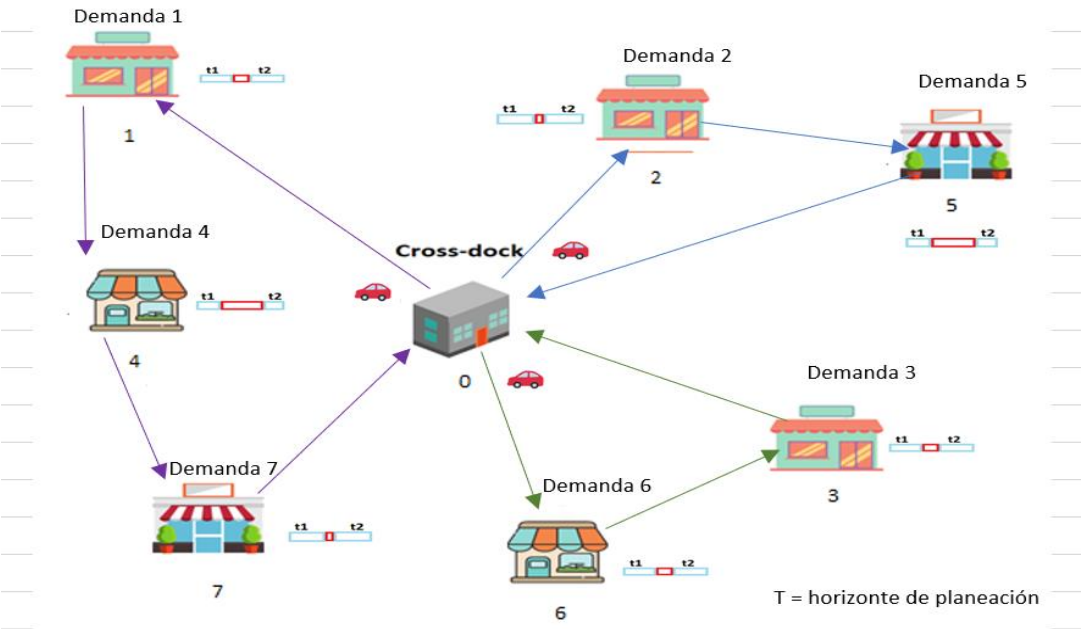


Figura 3.

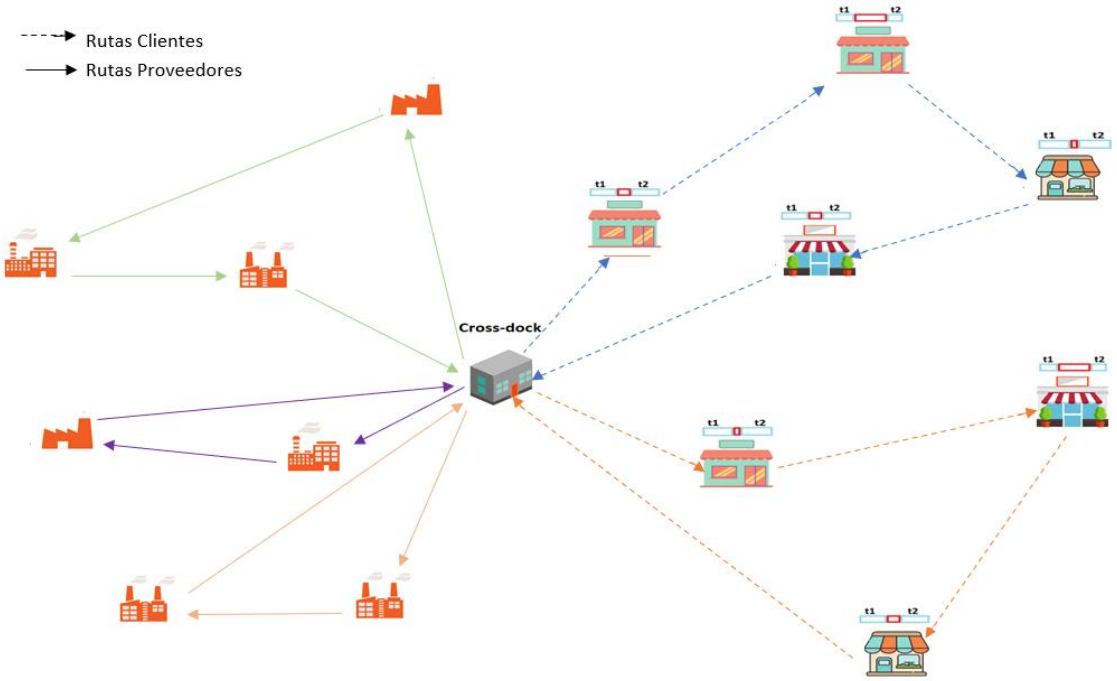
Representación gráfica del modelo para la parte 2.



La representación final del modelo se puede observar en la figura 4, la cual abarca un modelo de ruteo de vehículos con cross-dock y ventanas de tiempo, este problema contiene múltiples proveedores de capacidades diferentes especializados en el mismo tipo de producto, un centro de cross-dock y múltiples minoristas con demandas distintas y sus respectivas ventanas de tiempo.

Figura 4.

Representación gráfica del modelo.



1.3.3. Etapa 3: Programación de los algoritmos (Dando cumplimiento al objetivo 3).

- Estudiar las herramientas y el lenguaje de programación para Matlab.
- Programar los algoritmos para la parte 1 y 2 respectivamente de la heurística adaptada del vecino más cercano en el lenguaje de programación para Matlab, con el fin de obtener las soluciones iniciales que se usarán en los algoritmos de búsqueda tabú.

- Programar dos algoritmos de búsqueda tabú, uno para la primera parte y otro correspondiente a la segunda en el lenguaje de programación para Matlab.

1.3.4. Etapa 4: Prueba y validación de los algoritmos (Dando cumplimiento al objetivo 4).

- Efectuar análisis estadísticos que permitan evaluar la eficiencia del algoritmo búsqueda tabú en diferentes condiciones.
- Evaluar los algoritmos con instancias adaptadas de la literatura.
- Comparar los resultados del algoritmo búsqueda tabú con los del vecino más cercano y evaluar en qué medida es mejor el uno del otro.

1.3.5. Etapa 5: Documentar resultados (Dando cumplimiento al objetivo 5).

- Consolidar y sintetizar los resultados presentados en un libro y artículo de carácter publicable.

2. Revisión de la literatura

2.1. Revisión sistemática de la literatura.

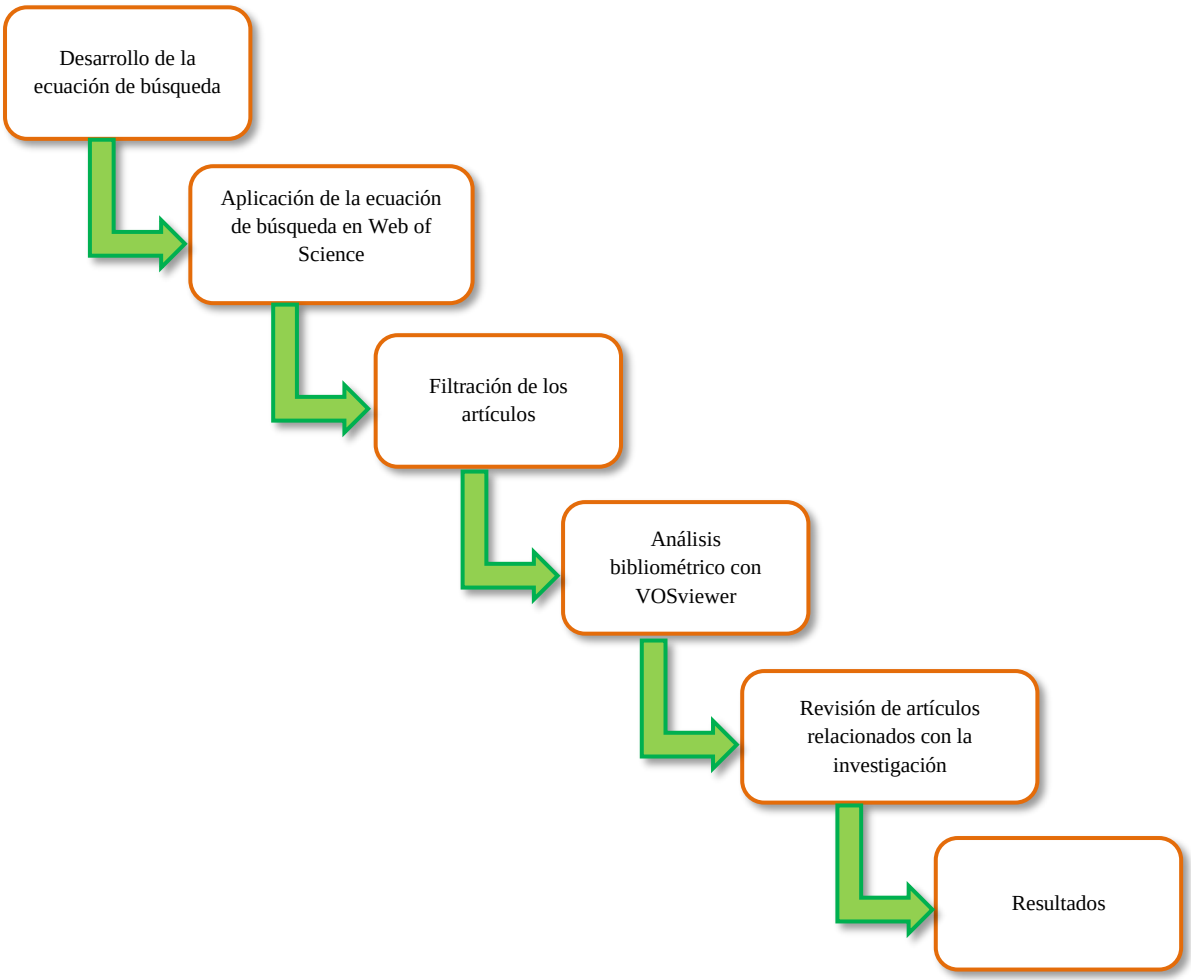
En primera instancia, para el desarrollo de esta investigación se optó por buscar artículos y documentos que ampliaran el panorama y brindaran información sobre la temática del Ruteo de vehículos (VRP) y problema del Agente viajero (TSP), una vez consultada la información referente a estas temáticas y las diferentes variantes que se presentan en el VRP se hizo la elección de la problemática a desarrollar en esta investigación, la cual era el problema de ruteo de vehículos con ventanas de tiempo añadiendo el concepto de cross-docks.

De esta manera se obtuvo una aproximación de la utilidad que tiene esta temática en la optimización de la cadena de suministro en las empresas distribuidoras y cómo se pueden

disminuir los costos logísticos de éstas, además de identificar los principales métodos de estudio de dichas investigaciones, sus aplicaciones en la industria, algoritmos utilizados y resultados obtenidos. En la *figura 5* se presenta la secuencia de pasos adoptados para el cumplimiento de esta fase.

Figura 5.

Pasos de la revisión de literatura



Para la elaboración de la ecuación de búsqueda se tomaron en primer lugar los tres términos principales de la investigación los cuales son “vehicle routing problem”, “time window” y “cross-dock”, Luego se decide incluir diversos sinónimos de estos mismos que se usan en la literatura tales como “location problem” y “time depended”, del mismo modo, resulta

oportuno agregar las diversas maneras en las se escribe el término “cross-dock”. Por último, Al haber escogido como algoritmo la búsqueda tabú, se decide añadir esta última expresión como “tabú search”, al finalizar quedó la ecuación de búsqueda que se muestra en la *figura 6.*

Figura 6.

Ecuación de búsqueda

```
TS=(("vehicle* routing* problem*" OR "location problem") AND ("cross dock*" OR
crossdock* OR cross-dock*) AND ("time window*" OR "time depend") OR ("tabu
search") )
```

Al aplicar la Ecuación de búsqueda en la base de datos Web Of Science proporcionada por los recursos electrónicos de la Universidad Industrial de Santander se encontraron un total de 6.204 artículos los cuales fueron refinados para que la búsqueda priorizara aquellos artículos en los que el cross-dock fuera un foco importante, lo que permitió que la búsqueda se redujera a un total de 50 artículos con corte a diciembre de 2020, de los cuales 35 se tomaron en cuenta para realizar la revisión de literatura, dado que en estos se enfocan los temas concernientes al problema de ruteo de vehículos con ventanas de tiempo y cross-docking, la totalidad de artículos disponibles se encuentra en el **Apéndice A.**

Posteriormente, con el fin de conocer la incidencia de esta temática de investigación, se realizó un análisis bibliométrico por medio de la herramienta VOSviewer en donde se ingresaron la totalidad de los artículos refinados de la ecuación de búsqueda, teniendo en cuenta la densidad de artículos por año que se han venido desarrollando de este tema, los países que más han contribuido con investigaciones de este tipo, las palabras clave más utilizadas en estas y los autores que más han aportado en el tema, se puede acceder al análisis por medio del **Apéndice B.**

2.2. Análisis preliminar de la literatura

El problema de ruteo de vehículos (VRP) es un tema crucial en el transporte, la logística y la gestión de la cadena de suministro (Nosrati & Arshadi Khamseh, 2020), ya que por medio de éste se optimizan las rutas de distribución minimizando costos y reduciendo tiempos de respuesta a los clientes, en donde los vehículos abandonan un depósito, atienden a los clientes de la red y regresan al depósito después de completar sus rutas (Montoya-Torres et al., 2015). "Los problemas de VRP fueron investigados por primera vez en Tarantilis y Kiranoudis (2001), Hsu et al. (2007) y Kang y Lee (2007)" (Utama et al., 2020) y con el tiempo estos han ido evolucionando conforme las necesidades de la industria, haciendo los problemas cada vez más ajustados a la realidad, en donde se tienen en cuenta factores tales como las ventanas de tiempo, los diferentes sistemas de inventario, las capacidades de los depósitos, los vehículos, entre otros.

Un problema de ruteo de vehículos con demanda estocástica (distribución aleatoria), es aquel en el que un vehículo debe atender la demanda incierta de un conjunto de clientes, se debe asignar una ruta en donde se minimice la distancia recorrida esperada, haciendo que el costo de transporte sea mínimo. En el artículo de (Bianchic et al., 2004) se abordan diferentes meta-heurísticas para resolver el problema anteriormente mencionado comparando los resultados obtenidos por los algoritmos realizados con diferentes instancias de la literatura y verificando su eficiencia.

Teniendo en cuenta que puede salir costoso el hecho de que un camión solo atienda a un cliente y proveedor, (Yazdani et al., 2017) abordan un problema de programación de vehículos y de rutas para que de esta manera un se pueda atender a varios clientes/proveedores en el proceso de recogida/entrega del producto tratando con meta-heurísticas tales como una búsqueda tabú y un algoritmo genético para la solución de este planteamiento.

Un factor importante en los problemas de ruteo de vehículos son los tiempos de viaje, ya que estos por lo general tienen como objetivo principal minimizarlos, para lograr esto satisfactoriamente hay que tener en cuenta diferentes limitantes, una de ellas es el tiempo en el cual los clientes pueden ser atendidos, a esto se le llama ventana de tiempo y por lo general es exigida por el cliente.

En el artículo escrito por (Restrepo et al., 2008) se aborda un problema de ruteo de vehículos con ventanas de tiempo las cuales se fijaron como las fechas prometidas a los clientes, tiene como objetivo principal minimizar los costos de transporte y se debe asignar el orden en el que cada vehículo debe visitar a los respectivos clientes usando la heurística R.

En el documento realizado por (Chen et al., 2011) se aborda un problema de ruteo de vehículos con depósitos virtuales y ventanas de tiempo, donde se tiene en cuenta que cuando estos van en una ruta pueden recargar los productos desde depósitos físicos o virtuales antes de continuar. Se debe cumplir el objetivo de minimizar el costo total del viaje teniendo en cuenta que cada cliente debe recibir el producto dentro de una ventana de tiempo fuerte, que un vehículo pequeño tiene capacidad limitada mientras que a los grandes no se les considera esta última y que el tiempo de trabajo de cada conductor no debe excederse del máximo tiempo de trabajo preestablecido.

Por otro lado, (Rashidi Komijan & Delavari, 2017), consideran un caso real de una distribuidora de alimentos (Amirpakhsh Company) que entrega y recibe productos perecederos en diferentes áreas, donde el objetivo de este es modelar un VRP para minimizar costos totales, teniendo en cuenta las ventanas de tiempo exigidas considerando el tipo de producto que distribuye. Para la solución de este problema se usa Gurobi solver en GAMS.

Otro factor importante a tener en cuenta para la gestión de la cadena de suministro, reducir costos e incrementar la satisfacción de los clientes, es la implementación de cross

docking, pues este es de vital importancia al momento de reducir inventarios teniendo en cuenta que al usar esta metodología se asegura que el tiempo de almacenamiento del producto sea el menor posible.

En el trabajo de (Lee et al., 2006), se aborda un problema de ruteo de vehículos cuyo objetivo es determinar la cantidad de vehículos junto a la mejor ruta, horario y hora de llegada para cada uno de los coches en un cross-dock y de esta manera minimizar los costos de transporte proponiendo un algoritmo heurístico basado en un algoritmo de búsqueda tabú.

(Wen et al., 2009) en su artículo, tratan un problema de ruteo de vehículos homogéneos con cross-dock (VRPCD) para llevar un producto de los proveedores a los clientes pasando por un cross-dock con el objetivo de minimizar los tiempos de viaje teniendo en cuenta las ventanas de tiempo y la duración esperada de toda la operación de transporte.

En el estudio de (Yin & Chuang, 2016) se plantea un nuevo algoritmo a un problema de VRPCD previamente estudiado planteando este con vehículos ecológicos, de tal manera que se tienen en cuenta tanto los costos y otros factores, tales como las emisiones de carbono emitidas por lo vehículos.

El problema de VRPCD tratado por (Wang et al., 2017) cuenta con entregas divididas y se asignan los vehículos a los cross-docks optimizando el orden de visita de cada camión a los proveedores y los clientes combinando una heurística constructiva con recocido simulado de dos capas y la búsqueda tabú.

Así mismo (Ladier & Alpan, 2018) proponen programar las llegadas y salidas de los camiones que entran y salen del cross-dock con unas ventanas de tiempo para cada cliente, al mismo tiempo que se planifica la manipulación interna de pallets, teniendo en cuenta las

ventanas de tiempo, con el objetivo de minimizar tanto el número de pallets almacenados, como la insatisfacción de los proveedores.

(Ozden & Saricicek, 2019) abordan un problema de asignación de vehículos en un sistema de cross-docking. Para ello, en su estudio consideran tres tipos de camiones, los cuales son: de entrada, salida y compuestos (entrada y salida), los cuáles deben ser asignados a un cross-dock de múltiples puertas, también se centra en hallar la secuencia de llegada de dichos vehículos teniendo en cuenta las ventanas de tiempo establecidas, siendo el objetivo del problema minimizar las tardanzas y las salidas apresuradas de los camiones proponiendo para su solución algoritmos de recocido simulado y búsqueda tabú.

Por último (Hasani Goodarzi et al., 2020) en su artículo abordan un problema de ruteo de vehículo con cross docking y ventanas de tiempo (VRPTWCD), donde propone un modelo de optimización y un algoritmo de solución dándole cabida a dos objetivos en conflicto los cuales son minimizar el costo total operativo y la suma de la máxima anticipación y tardanza.

3. Marco de Referencia

3.1. Marco teórico

3.1.1. Problema del agente viajero

El problema del agente viajero (TSP) ha estado en la mira de los matemáticos e informáticos por su facilidad para describirlo y entenderlo, pero de gran dificultad para resolver. El problema puede describirse de la siguiente manera: un vendedor ambulante ubicado en una ciudad A desea visitar cada una de las n-1 ciudades una sola vez y regresar a la ciudad de origen de manera tal que haya visitado todas las ciudades y a su vez haber recorrido la menor distancia posible (donde el costo de viajar de una ciudad *i* a la ciudad *j* está definido por *Cij*) (Daza et al., 2009). En una búsqueda de minimizar el costo total del recorrido el TSP

actúa como la base para problemas de optimización combinatoria como lo viene a ser el ruteo de vehículos.

3.1.2. Problema de ruteo de vehículos

El problema de ruteo de vehículos (VRP) es una modificación que amplía el rango de acción y práctica propuesto por el TSP. Los ejercicios de VRP son problemas bastante estudiados y conocidos en el área de la investigación de operaciones y la optimización combinatoria debido a la dificultad que implica resolver este tipo de ejercicios, debido a lo complejo que resulta hallar una solución óptima en ejercicios de gran tamaño. Para esta modalidad, los problemas VRP están bajo la clasificación NP-hard (Nagy & Salhi, 2006). El VRP es un problema de optimización en el que un grupo de destinos (clientes, proveedores, depósitos, etc.) deben ser visitados por una flota de vehículos de reparto los cuales deben satisfacer sus respectivas demandas (Morais et al., 2014). Debido al gran interés y cantidad de estudios de este tipo de ejercicios han surgido otras variantes del VRP motivadas por las diferentes aplicaciones que se pueden dar en la vida real, entre estas aplicaciones se pueden citar, por ejemplo. VRP con ventanas de tiempo (VRPTW), VRP con vehículos capacitados (CVRP), VRP con flota heterogénea, VRP con depósitos múltiples (MDVRP), VRP con recolección y entrega (VRPPD), VRP con centros de cross-docking, etc.

3.1.3. Optimización combinatoria

Los problemas de optimización combinatoria son una serie de problemas que tienen como principales características su facilidad al momento de plantearlos y elevada dificultad de resolución. Algunos ejemplos de optimización combinatoria son el problema de la mochila, la coloración de grafos, el agente viajero y el ruteo de vehículos. Estos problemas tienen por objetivo encontrar el valor máximo (o mínimo dependiendo del caso) de una función objetivo (F) sobre un conjunto finito de soluciones indicado como (S), el cual al ser finito hará que las

variables resulten en general discretas, restringiendo su dominio a una serie finita de valores (Schweickardt et al., 2010). La optimización combinatoria al tener un gran número de elementos (S) hace inviable la evaluación de todas las soluciones para determinar la óptima, lo que supone resulta más factible resolver mediante un método heurístico que la utilización de un procedimiento exacto para su resolución.

3.1.4. Teoría de la complejidad computacional

La teoría de la complejidad computacional forma parte de la teoría de la computación, esta se encarga de estudiar los recursos que se necesitan durante un cálculo para dar solución a un problema, de manera tal que un cálculo se define como complejo si es difícil de realizar (Cortez, 2004). De este modo, la complejidad de cálculo se define como la cantidad de recursos computacionales que se requieren para efectuar un cálculo.

En la complejidad computacional se presentan dos tipos de problemas que son P y NP, los cuales abarcan los problemas de tiempo de resolución polinomial determinístico y no determinístico. De este modo, se dictamina que los problemas P tienen solución eficiente, mientras que los problemas NP son más difíciles de resolver (Rosenfeld & Irazábal, 2020). De igual manera se han definido varias clases de problemas que son las siguientes:

- Clase P: Comprende los problemas que pueden ser resueltos por un programa en un tiempo relativamente rápido, de manera tal que la resolución de estos dentro de un algoritmo en una cantidad limitada de pasos teniendo un comportamiento polinomial.

- Clase NP: Esta clase tiene mayor complejidad, pues no solo abarca todos los problemas de la clase P, también abarca problemas como, por ejemplo: Diseño de circuitos, planificación del trabajo, etc. Por otro lado, a pesar de tomar un tiempo computacional mayor, la clase NP se caracteriza en que las soluciones a sus respectivos problemas

pueden ser verificadas en un tiempo polinómico por medio del uso de una máquina de Turing no determinista.

- Clase NP-hard: Son los problemas NP con mayor complejidad, pues estos tienen la característica de que los tiempos de solución y verificación crecen exponencialmente a medida que aumenta el tamaño del problema, llegando incluso a tener una complejidad desconocida, además son problemas que se resuelven en un tiempo polinomial no determinista. Algunos ejemplos de problemas NP-hard son: el ruteo de vehículos y el ajedrez.

- Clase NP-completo: Son problemas que tienen mayor dificultad que la clase NP, pues a diferencia esta, resulta menos probable que los problemas correspondan a la clase P, lo que limita la posibilidad de resolver el problema rápidamente en un tiempo polinómico.

3.1.5. Cross-docking

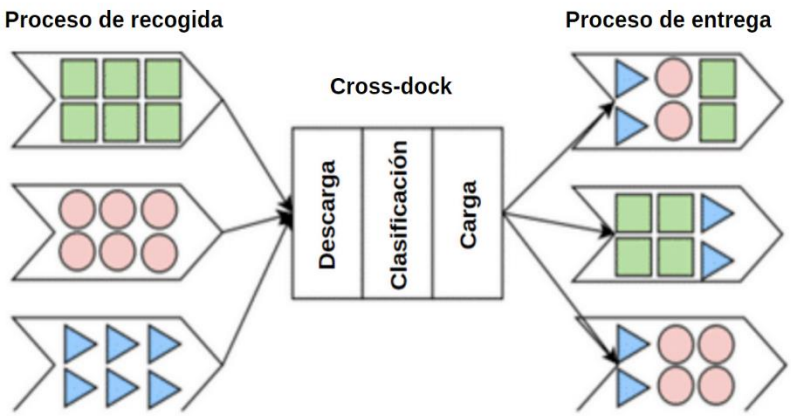
El cross-Docking (CD) “es un concepto de manipulación y distribución de productos en el que los artículos del producto se clasifican y reconocen en función de las demandas del cliente y se mueven directamente desde el muelle de recepción al muelle de envío” (Mohtashami, 2015), asimismo, el CD es conocido como una de las principales estrategias de distribución a nivel logístico dentro de una cadena de suministro. Buijs et al (2014) señalan que el CD tiene por objetivo reducir considerablemente los costos de transporte, mano de obra, acelerar el flujo de productos y los tiempos de entrega, esto último por medio de la eliminación o reducción en los tiempos de almacenamiento y selección de pedidos en un almacén (Esmizadeh & Mellat Parast, 2020). Van Belle y Col (2012) afirman que “Esta estrategia aporta diferentes ventajas frente a los centros de distribución tradicionales: consolidación de envíos, menor tiempo de entrega, reducción de costes, mejor servicio al cliente, menor exceso de

existencias, etc." (citado en Küçükoğlu & Öztürk, 2019). Schwind (1995) describió que el CD juega un papel importante en la eliminación de existencias adicionales y el almacenamiento propio de los almacenes tradicionales (citado en Ahkamiraad & Wang, 2018).

El cross-docking es una estrategia que resulta de vital importancia dependiendo del sector al que pertenezca la organización, sirviendo como ejemplo aquellas empresas que se dediquen al e-commerce, en especial cuando las operaciones logísticas están sujetas a descuentos y rebajas. Otros ejemplos son el sector farmacéutico y el sector alimenticio, ya que en ocasiones es necesario controlar la temperatura de ciertos medicamentos junto a la exigencia de suministro de stock rápido, en el caso de los alimentos es de gran importancia para aquellas empresas que trabajan con alimentos perecederos. En la *figura 7* se puede observar una representación gráfica del proceso de distribución cuando se cuenta con un centro de cross-dock.

Figura 7.

Distribución de productos por medio de un cross-dock



Van Belle, Valckenaers y Cattrysse (2012) señalan que los estudios realizados en el campo de cross-docking se pueden clasificar de la siguiente manera: ubicación de los cross-

dock, asignación de camiones, programación de camiones y programación de recursos en el cross-dock (citado en Moghadam et al., 2014).

3.1.5.1. Centro de cross-docking. Un centro de CD es un tipo de almacén que permite preparar los pedidos de los clientes sin la necesidad de pasar por la fase de almacenamiento para cada uno de los productos y posteriormente prepararlos para su entrega. En este tipo de almacenes los bienes llegan en camiones entrantes los cuales previamente han recogido la mercancía suministrada por los diferentes proveedores y la dejan en las puertas de entrada, acto seguido los productos se clasifican y consolidan según su destino y por medio de un proceso de manipulación el cual hace uso de dispositivos como carretillas elevadoras permite que estos se transfieren directamente a las puertas de salida de la instalación, donde posteriormente se cargan en los camiones de salida los cuales se encargarán de distribuir los productos y hacer la entrega al cliente final (Gelarech et al., 2020).

3.1.5.2. Problema de ruteo de vehículos con cross-dock. El problema de ruteo de vehículos con cross-dock (VRPCD) tiene por objetivo esencial determinar la cantidad de vehículos, la mejor ruta, horario y hora de llegada para cada vehículo en un CD para minimizar el costo de transporte, considerando el cross-docking en el horizonte de planificación T (Lee et al., 2006). El VRPCD tiene como característica que funciona como dos problemas de ruteo de vehículos diferentes que se conectan en el CD para formar un solo ejercicio, sin embargo, se pueden presentar pequeñas variaciones como asignar múltiples centros de cross-docking, vehículos heterogéneos para los proveedores y clientes, etc.

3.1.6. Ventana de tiempo

Una ventana de tiempo (TW) es el intervalo de tiempo que abarca el tiempo mínimo y máximo que un cliente permitirá para la entrega de los productos en horarios previamente definidos (Desrochers et al., 1992). En el ruteo de vehículos las ventanas de tiempo pueden tener otras variaciones con el fin de hacerlas más cercanas a la realidad, estas variaciones son: ventanas de tiempo suaves y duras. En las ventanas de tiempo suaves el producto se puede entregar fuera de los intervalos de tiempo definidos por el cliente a cambio de pagar un extra monetario. Por otro lado, las ventanas de tiempo duras no permiten que se genere ningún incumplimiento en los horarios de atención establecidos por el minorista o cliente y estos horarios pueden ser diferentes entre los diferentes periodos contemplados en el horizonte de planeación (Pérez & Guerrero, 2015).

3.1.6.1. Ruteo de vehículos con ventanas de tiempo. El problema de ruteo de vehículos con ventanas de tiempo (VRPTW) es una variante del VRP que consiste básicamente en minimizar los costos de transporte sujetos a restricciones de tiempo y capacidad para cada ruta con base a la demanda de cada cliente (Cruz-Chávez & Díaz-Parra, 1999). El VRPTW se puede formular usando ventanas de tiempo suaves o duras dependiendo de las necesidades del problema propuesto. No obstante, primero se deben contemplar las restricciones de capacidad de los vehículos para su planificación en las entregas, ya que muchas compañías logísticas dedicadas al transporte y entrega de productos poseen una gran cantidad de pedidos con tiempos de entrega definidos por sus clientes, motivo por el cual la utilización de las ventanas de tiempo se convierte en una necesidad innegable para el planteamiento del modelo.

3.1.7. Problema de ruteo de vehículos con ventanas y tiempo y cross-dock

El problema de ruteo de vehículos con ventanas de tiempo y cross-dock (VRPTWCD) planteado por Lee et al (2006) busca asignar recorridos a un conjunto de vehículos en el CD para que los proveedores y los clientes sean visitados dentro de las respectivas ventanas de tiempo (Wen et al., 2009). Igualmente, debido a la extensión de aplicaciones que puede tener este problema se pueden presentar múltiples adiciones que complementen el problema original planteado por Lee et al (2006).

3.1.8. Heurística

Una heurística es cualquier dispositivo (programa, estructura de datos, estrategia, conocimiento, etc.), que se utiliza para la resolución de problemas, la cual no asegura llegar a una solución óptima, pero que ofrece razones suficientes para creer que será útil y que se agregará en un sistema de resolución de problemas con la expectativa de que, en promedio, el rendimiento mejorará. (Romanycia & Pelletier, 1985).

3.1.8.1. Método heurístico.

> “Un méto do heurístico constituye unprocedimiento para resolver un problema de optimización bien definido, mediante una aproximación intuitiva, en la que la estructura del problema se utiliza de forma inteligente para obtener una buena solución.” (Schweickardt et al., 2010).

3.1.9. Problema del vecino más cercano

El método del vecino más cercano es un procedimiento heurístico que pretende viajar al destino con la distancia más corta partiendo de un nodo actual en el que se encuentra el viajante.

3.1.10. Metaheurística

Una metaheurística se define como un proceso de generación iterativo compuesto por estrategias de alto nivel que guían una heurística subordinada mediante la combinación de diferentes conceptos que permitan la exploración y explotación del espacio de búsqueda, permitiendo escapar de óptimos locales por medio del uso de un conjunto estrategias de aprendizaje las cuales van desde simples procesos de búsqueda local hasta procesos de aprendizaje relativamente complejos que permiten estructurar la información y producir eficientemente soluciones de alta calidad. (Blum & Roli, 2003).

3.1.10.1. Algoritmo metaheurístico. Los algoritmos metaheurísticos son algoritmos aproximados de optimización compuestos por procedimientos iterativos, los cuales buscan combinar lo básico de los métodos heurísticos en marcos de nivel superior de forma inteligente, aprovechando distintos conceptos que tienen como propósito explorar un espacio de búsqueda de manera más efectiva (Blum & Roli, 2003). Algunos ejemplos de algoritmos metaheurísticos son, por ejemplo, la búsqueda tabú (TS), algoritmos genéticos (GA), colonia de hormigas (ACO), búsqueda local iterada (ILS), etc.

3.1.11. Algoritmo de búsqueda Tabú

La búsqueda tabú (TS) fue propuesta por Fred Glover en 1986, como parte de un algoritmo metaheurístico iterativo que guía un procedimiento de búsqueda local para explorar el espacio de la solución permitiendo escapar de óptimos locales y llegar a una mejor alternativa. (Fred Glover, 1989). La TS mejora el rendimiento de un método de búsqueda local mediante el uso de estructuras que poseen memoria, una vez se haya determinado una posible solución esta será marcada como tabú, para que el algoritmo no vuelva a buscar esa posibilidad (Vahdani et al., 2012).

Landrieu et al (2001) señalan que el algoritmo TS se caracteriza por tener un mecanismo de memoria denominado “lista tabú” el cual evita que la búsqueda vuelva a caer en soluciones visitadas previamente para un número determinado de iteraciones las cuales se almacenan en un historial de búsqueda el cual conserva los movimientos realizados o atributos obtenidos en un plazo de iteraciones definido (citado en Küçükoğlu & Öztürk, 2019).

Por otro lado, si la solución resulta lo suficientemente atractiva al momento de la búsqueda la restricción puede llegar a ignorarse, este procedimiento recibe el nombre de criterio de aspiración y permite activar una excepción en la lista tabú siempre y cuando el resultado lleve a una mejor respuesta (Armentano & Yamashita, 2000).

La búsqueda tabú busca evitar caer en ciclos mientras ejecuta una técnica de búsqueda local, un complemento de esta última es la búsqueda tabú reactiva en la cual el tamaño de la lista tabú se adapta para considerar más a fondo el problema de optimización (Yazdani et al., 2017).

4. Descripción del problema y formulación del modelo matemático

En esta sección se define el problema de ruteo de vehículo con ventanas de tiempo y un centro de Cross-dock. El modelo presentado en este documento se basa en el propuesto para la problemática que se encuentra en (Ahkamiraad & Wang, 2018).

Para el planteamiento del problema del VRPTWCD se optó por dividir este en dos partes, la primera corresponde a las rutas de recogida que trata la mercancía suministrada por los proveedores, y la segunda está enfocada a las rutas de entrega de los clientes, donde cada parte tendrá su respectiva función objetivo, y la unificación de los resultados factibles u óptimos de estas dos funciones corresponden a la solución general del problema de VRPTWCD.

4.1. Parte 1 del problema

4.1.1. Descripción del problema

La parte uno del problema, correspondiente a las rutas de recogida de mercancía con cada uno de los proveedores, se abordará por medio de un problema ruteo de vehículo con cross dock que se encuentra conformado por una flota de vehículos provenientes del centro de cross-dock, los cuales recogerán toda la mercancía ofrecida por los proveedores y se devolverán simultáneamente al punto de origen (cross-dock).

Supuestos del modelo para la parte 1.

- La mercancía suministrada por los proveedores será la suficiente para satisfacer todos los clientes del problema.
- La cantidad de mercancía recogida será exactamente igual a la entregada, por lo que al finalizar el problema no habrá inventario sobrante.
- los vehículos utilizados para atender los proveedores son capacitados, es decir, tiene capacidad limitada.
- La capacidad de los vehículos de recogida puede ser mayor a la de los vehículos de entrega.
- Para la parte 1 del modelo no hay horizonte de planeación, es decir, los tiempos transcurridos de las rutas para la parte 1 no tiene importancia en la parte 2.
- Los vehículos parten del cross-dock y deben volver al mismo una vez finalizado el ejercicio.
- Todos vehículos entrantes llegan al cross-dock al mismo tiempo para realizar el transbordo en el cross-dock hacia los vehículos de entrega.
- El costo de viajar entre nodos es equivalente a la distancia recorrida y la relación entre estos dos parámetros es de uno a uno, es decir $c_{ij} = d_{ij}$.

- El nodo 0 corresponde al cross-dock
- Los costos del problema son simétricos, es decir, el costo de viajar del nodo i al nodo j es el mismo de si se viajara en sentido contrario.
- El cross-dock deberá atender todos los proveedores disponibles.
- El tipo de mercancía suministrada por los proveedores es homogéneo, es decir que ofrecen un mismo tipo de producto y no se tomarán en cuenta variables para diferenciar el tipo de mercancía en este modelo.
- Un vehículo no puede atender más de una ruta.

4.1.2. modelo matemático

Índices del modelo

i: Nodo de partida.

j: Nodo de destino.

k: Número del vehículo asignado.

Parámetros

$\boldsymbol{L}$**:** Número máximo permitido de nodos en una ruta de recogida

$\boldsymbol{m}$**:** Número de vehículos de recogida en el cross-dock

$\boldsymbol{Q}$**:** Capacidad máxima de cada vehículo de recogida

$\boldsymbol{Y}$**:** Capacidad máxima del cross-dock

$\boldsymbol{C_{ijk}}$**:** Costo de viajar entre los nodos (i, j) con el vehículo k

$\boldsymbol{Cv_k}$**:** Costo fijo de operar el vehículo k

$\boldsymbol{t_{ij}}$**:** Tiempo necesario para viajar entre los nodos (i, j)

$\boldsymbol{V_i}$: Duración de la visita en el nodo i por un vehículo

$\boldsymbol{p}$**:** Nodo j que pertenece a un punto de recogida

p_j: Cantidad de mercancía recogida en el nodo j

Variables

l_{ik}**:** Tiempo en que sale un vehículo k del nodo i

u_i**:** Variable entera no negativa

u_j**:** Variable entera no negativa

X_{ijk}**:** Si el vehículo k pasa por el enlace (i, j) = 1, de lo contrario es cero

(1)

$$Min \sum_{i=0}^{n} \sum_{j=0}^{n} \sum_{k=1}^{m} C_{ijk} X_{ijk} + Cv_k$$

La ecuación (1) representa la función objetivo para el VRPTW de las rutas correspondientes a los clientes con el cross-dock.

(2)

$$\sum_{i=0}^{n} \sum_{k=1}^{m} X_{ijk} = 1, \qquad \forall j = \{1, 2, \dots, n\}, \qquad \forall k = \{1, 2, \dots, m\}, \qquad i \neq j$$

(3)

$$\sum_{j=0}^{n} \sum_{k=1}^{m} X_{ijk} = 1, \qquad \forall i = \{1, 2, \dots, n\}, \qquad \forall k = \{1, 2, \dots, m\}, \qquad i \neq j$$

Las restricciones (2) y (3) garantizan que la solicitud de cada nodo de recogida se satisfaga en una visita y solo por un vehículo.

(4)

$$\sum_{i=0}^{n} X_{ipk} - \sum_{j=0}^{n} X_{pjk} = 0\,, \qquad \forall p \in \{1, 2, \ldots, n\}, \qquad \forall k \in \{1, 2, \ldots, m\}, \qquad p \neq i,$$

$$j \neq p$$

La restricción (4) representa si un vehículo k entra en el nodo de recogida p, este finalmente debería salir del nodo.

(5)

$$\sum_{j=1}^{n} X_{0jk} \leq 1\,, \qquad \forall k \in \{1, 2, \ldots, m\}$$

(6)

$$\sum_{i=1}^{n} X_{i0k} \leq 1\,, \qquad \forall k \in \{1, 2, \ldots, m\}$$

Las restricciones (5) y (6) muestran que el nodo inicial y el nodo final de cada ruta debe ser el nodo 0, es decir el cross-dock.

(7)

$$\sum_{j=1}^{n} X_{0jk} = \sum_{i=1}^{n} X_{i0k}\,, \qquad \forall k \in \{1, 2, \ldots, m\}$$

La restricción (7) expresa que cada vehículo inicia su ruta en el cross-dock debe regresar a este.

(8)

$$\sum_{j=1}^{n} \boldsymbol{Pj} \leq \boldsymbol{Y}, \qquad \forall \boldsymbol{k} \in \{\boldsymbol{1}, \boldsymbol{2}, \dots, \boldsymbol{m}\},$$

La restricción (8) representa la limitación de capacidad del cross-dock

(9)

$$\sum_{j=1}^{n} \sum_{K=1}^{m} \boldsymbol{X_{0jk}} = \boldsymbol{m}$$

La restricción (9) muestra el total de vehículos de recogida en el cross-dock, evidenciando que existe un vehículo de recogida para cada proveedor en caso de ser necesario.

(10)

$$\sum_{i=0}^{n} \sum_{j=0}^{n} \boldsymbol{p_j}\, \boldsymbol{X_{ijk}} \leq \boldsymbol{Q}\,, \qquad \forall \boldsymbol{k} \in \{\boldsymbol{1}, \boldsymbol{2}, \dots, \boldsymbol{m}\}, \qquad \boldsymbol{i} \neq \boldsymbol{j}$$

La restricción (10) representa la limitación de capacidad para cada vehículo de recogida.

(11)

$$\boldsymbol{u_i} - \boldsymbol{u_j} + \boldsymbol{L} \sum_{k}^{m} \boldsymbol{X_{ijk}} \leq \boldsymbol{L} - \boldsymbol{1}, \qquad \forall \boldsymbol{i} \in \{\boldsymbol{1}, \boldsymbol{2}, \dots, \boldsymbol{n}\}, \qquad \forall \boldsymbol{k} \in \{\boldsymbol{1}, \boldsymbol{2}, \dots, \boldsymbol{m}\}$$

La restricción (11) garantiza que no hayan subrutas en el ejercicio, es decir que una ruta de entrega no inicie y finalice fuera del cross-dock.

(12)

$$\boldsymbol{ui \geq 0, donde\ "ui"\ es\ un\ entero}$$

$$\boldsymbol{uj \geq 0, donde\ "uj"\ es\ un\ entero}$$

La restricción (12) asegura que solo se puedan asignar valores no negativos a las variables enteras *ui* y *uj*.

(13)

$$\boldsymbol{X_{ijk}} \begin{cases} \mathbf{1}, & \boldsymbol{Si\ el\ vehículo\ k\ pasa\ por\ el\ enlace\ (i,j)} \\ \mathbf{0}, & \boldsymbol{sino} \end{cases}$$

$$\boldsymbol{X_{ijk} \in \{0, 1\}}$$

La restricción (13) describe la restricción binaria de la variable de decisión.

4.2. Parte 2 del problema

4.2.1. Descripción del problema

La parte dos del problema, corresponde a la ruta de entrega de la mercancía a cada uno de los clientes, la cual se abordará por medio de un problema de ruteo de vehículos con ventanas de tiempo (VRPTW), el cual se encuentra conformado por una serie de vehículos pertenecientes al cross-dock, este se encargará de enviar el producto a todos los clientes de la ruta, para que finalmente todos los vehículos regresen al origen (cross-dock) sin mercancía sobrante.

Supuestos del modelo para la parte 2

- Los vehículos de entrega salen del cross-dock con la cantidad de mercancía necesaria para atender a todos los clientes de su ruta sin exceder sus límites de capacidad.
- Al finalizar el problema no habrá inventario sobrante.
- Los vehículos utilizados para atender los clientes son capacitados, es decir, tienen capacidad limitada y pueden ser de menor tamaño que los de recogida utilizados la parte 1, por lo que su capacidad y costos fijos de operación pueden ser diferentes.
- Ninguna ruta podrá pasarse del horizonte de planeación establecido.
- Cada vehículo que parte del cross-dock, debe volver al mismo una vez finalizado el ejercicio.
- El número de vehículos no se considera una limitante de peso en este problema, ya que se cuenta con una flota muy grande de vehículos, de manera que es posible tomar la peor decisión posible. Dicho en otras palabras, que cada cliente pueda ser atendido de forma exclusiva por un vehículo.
- El costo de viajar entre nodos es equivalente a la distancia recorrida y al tiempo de viaje entre los mismos (excluyendo las esperas para que inicien ventanas de tiempo y los tiempos de servicio), y la relación entre estos tres parámetros es de uno a uno, es decir $c_{ij} = d_{ij} = t_{ij}$.
- El nodo 0 corresponde al cross-dock
- Los costos del problema son simétricos, es decir, el costo de viajar del nodo i al nodo j es el mismo de si se viajara en sentido contrario.
- El cross-dock deberá satisfacer todos los clientes del ejercicio, dicho en otras palabras, la capacidad del cross-dock es lo suficientemente grande para recibir

toda la mercancía traída por los vehículos de recogida de la parte 1 y trasladarla a los vehículos de entrega para atender a los clientes.

- El tipo de mercancía entregada a los clientes es homogénea, es decir se ofrece un mismo tipo de producto y no se tomarán variables que permitan diferenciar el tipo de mercancía en este modelo.
- La cantidad de vehículos programados en el modelo es igual a la cantidad de las rutas totales de entrega.
- Los tiempos de permanencia dentro de cada nodo son insignificantes, es decir, toman valor de cero.
- Un vehículo no puede atender más de una ruta.

4.2.1 modelo matemático

Índices del modelo

i: Nodo de partida.

j: Nodo de destino.

k: Número del vehículo asignado.

Parámetros

$\boldsymbol{L}$**:** Número máximo permitido de nodos en una ruta de entrega

$\boldsymbol{T}$**:** Horizonte de planeación

$\boldsymbol{m}$**:** Número de vehículos de entrega en el cross-dock

$\boldsymbol{Q}$**:** Capacidad máxima de cada vehículo de entrega

$\boldsymbol{Y}$**:** Capacidad máxima del cross-dock

$\boldsymbol{C_{ijk}}$**:** Costo de viajar entre los nodos (i, j) con el vehículo k

$\boldsymbol{Cv_k}$**:** Costo fijo de operar el vehículo k

t_{ij}**:** Tiempo necesario para viajar entre los nodos (i, j)

[a_i, b_i**]:** Ventana de tiempo para salir del nodo i

d**:** nodo j que pertenece a un punto de entrega

d_j: cantidad de mercancía descargada en el nodo de entrega.

v_j: Tiempo de servicio en el nodo j

Variables

L_{ik}**:** Tiempo en que sale un vehículo k del nodo i

L_{jk}**:** Tiempo en que sale un vehículo k del nodo j

l_{jk}**:** Tiempo en que entra un vehículo k al nodo j

r_k**:** Tiempo de llegada del vehículo k al cross-dock (final de la ruta de ese vehículo)

u_i**:** Variable entera no negativa

u_j**:** Variable entera no negativa

X_{ijk}**:** Si el vehículo k pasa por el enlace (i, j) = 1, de lo contrario es cero.

(1)

$$Min \sum_{i=0}^{n} \sum_{j=0}^{n} \sum_{k=1}^{m} C_{ijk} X_{ijk} + Cv_k$$

La ecuación (1) representa la función objetivo para el VRPTW de las rutas correspondientes a los clientes con el cross-dock.

(2)

$$\sum_{i=0}^{n} \sum_{k=1}^{m} X_{ijk} = 1, \qquad \forall j = \{1, 2, \dots, n\}, \qquad \forall k = \{1, 2, \dots, m\}, \qquad i \neq j$$

(3)

$$\sum_{j=0}^{n}\sum_{k=1}^{m} X_{ijk} = 1, \qquad \forall i = \{1, 2, \dots, n\}, \qquad \forall k = \{1, 2, \dots, m\}, \qquad i \neq j$$

Las restricciones (2) y (3) garantizan que la solicitud de cada nodo de entrega se satisfaga en una visita y solo por un vehículo.

(4)

$$\sum_{i=0}^{n} X_{idk} - \sum_{j=0}^{n} X_{djk} = 0\,, \qquad \forall d \in \{1, 2, \dots, n\}, \qquad \forall k \in \{1, 2, \dots, m\}, \qquad d \neq i,$$

$$j \neq d$$

La restricción (4) representa si un vehículo k entra en el nodo de entrega d, este finalmente debería salir del nodo.

(5)

$$\sum_{j=1}^{n} X_{0jk} \leq 1\,, \qquad \forall k \in \{1, 2, \dots, m\}$$

(6)

$$\sum_{i=1}^{n} X_{i0k} \leq 1\,, \qquad \forall k \in \{1, 2, \dots, m\}$$

Las restricciones (5) y (6) muestran que el nodo inicial y el nodo final de cada ruta debe ser el nodo 0, es decir el cross-dock.

(7)

$$\sum_{j=1}^{n} X_{0jk} , = \sum_{i=1}^{n} X_{i0k} , \qquad \forall k \in \{1, 2, \ldots, m\}$$

La restricción (7) expresa que cada vehículo inicia su ruta en el cross-dock debe regresar a este.

(8)

$$\sum_{j=1}^{n} dj \leq Y, \qquad \forall k \in \{1, 2, \ldots, m\},$$

La restricción (8) representa la limitación de capacidad del cross-dock

(9)

$$\sum_{j=1}^{n} \sum_{K=1}^{m} X_{0jk} = m$$

La restricción (9) muestra el total de vehículos de entrega en el cross-dock, evidenciando que existe un vehículo de entrega para cada cliente en caso de ser necesario.

(10)

$$\sum_{i=0}^{n} \sum_{j=0}^{n} d_j X_{ijk} \leq Q , \qquad \forall k \in \{1, 2, \ldots, m\}, \qquad i \neq j$$

La restricción (10) representa la limitación de capacidad para cada vehículo de entrega.

(11)

$$a_i \leq l_{ik} \leq b_i, \qquad \forall i \in \{1, 2, \ldots, n\}, \qquad \forall k \in \{1, 2, \ldots, m\}$$

La restricción (11) representa la ventana de tiempo para cada nodo que se da en la restricción.

(12)

$$\sum_{i=0}^{n} \sum_{j=0}^{n} (t_{ij} + v_i) X_{ijk} \leq T, \qquad \forall k \in \{0, 1, 2, \ldots, m\}, \qquad i \neq j$$

La restricción (12) expresa que el tiempo total empleado en visitar los nodos y el transporte entre estos para cada una de las rutas atendidas por un vehículo k, debe ser menor o igual al horizonte de planeación *T*.

(13)

$$l_{jk} = \sum_{i=o}^{n} (t_{ij} + L_{ik})\, X_{ijk}, \qquad \forall k \in \{0, 1, 2, \ldots, m\}, \qquad \forall j \in \{1, 2, \ldots, n\}, \qquad i \neq j$$

La restricción (13) representa la siguiente relación: el tiempo en que llega el vehículo k al nodo j debe ser igual al tiempo de salida del nodo i más el tiempo necesario para viajar de i hasta j.

(14)

$$L_{jk} = \sum_{i=o}^{n} (t_{ij} + L_{ik})\, X_{ijk} + v_j, \qquad \forall k \in \{0, 1, 2, \ldots, m\}, \qquad \forall j \in \{1, 2, \ldots, n\}, \qquad i \neq j$$

La restricción (14) representa la siguiente relación: el tiempo en que sale el vehículo k del nodo j debe ser igual al tiempo de salida del nodo i más el tiempo necesario para viajar de i hasta j y el tiempo requerido para atender el nodo j (tiempo de servicio).

(15)

$$r_k = \sum_{i=1}^{n} (t_{i0} + L_{ik})\, X_{i0k}, \qquad \forall k \in \{1, 2, \dots, m\}$$

La restricción (15) modela la hora de llegada al cross-dock correspondiente a la ruta recorrida por el vehículo k.

(16)

$$u_i - u_j + L \sum_{k}^{m} X_{ijk} \leq L - 1, \qquad \forall i \in \{1, 2, \dots, n\}, \qquad \forall k \in \{1, 2, \dots, m\}$$

La restricción (16) garantiza que no hayan subrutas en el ejercicio, es decir que una ruta de entrega no inicie y finalice fuera del cross-dock.

(17)

$$ui \geq 0, donde\ "ui"\ es\ un\ entero$$

$$uj \geq 0, donde\ "uj"\ es\ un\ entero$$

La restricción (17) asegura que solo se puedan asignar valores no negativos a las variables enteras *ui* y *uj*.

(18)

$$X_{ijk} \begin{cases} 1, & Si\ el\ vehículo\ k\ pasa\ por\ el\ enlace\ (i,j) \\ 0, & sino \end{cases}$$

$$X_{ijk} \in \{0, 1\}$$

La restricción (18) describe la restricción binaria de la variable de decisión.

5. Programación del algoritmo

Debido a que el modelo matemático planteado anteriormente para el VRPTWCD es una variante del problema del agente viajero, el cual es un problema de gran complejidad computacional para su solución y que pertenece a la categoría de metaheurísticas, (Maldonado & Gómez, 2010) lo clasifican como un problema NP-Hard.

En esta investigación el problema planteado se resuelve usando una versión modificada del algoritmo TS que se planteó en "*Vehicle routing scheduling for cross-docking in the supply chain"* (Lee et al., 2006). En este articulo los autores proponen un método de solución al problema de ruteo de vehículos con cross-dock, donde para obtener la solución inicial se usó una heurística para definir un rango de tolerancia α con respecto a los costos de viaje, el cual actúa como filtro para una metaheurística que selecciona los menores costos. Posterior a este paso se implementa el algoritmo de búsqueda tabú para las etapas de recogida y entrega. Con base en la metaheurística anteriormente mencionada se construye una heurística que tiene como principio el criterio del vecino más cercano, en el que la solución encontrada se adapte a las restricciones del modelo para hallar la solución inicial y posteriormente implementarla en el algoritmo de búsqueda tabú para mejorar dicha solución.

En el presente capítulo se describe la heurística utilizada y el método de solución del TS en el problema de ruteo de vehículos con ventanas de tiempo y cross-dock.

5.1. Heurística adaptada del vecino más cercano

Para la ejecución del algoritmo de búsqueda tabú es necesario contar con una solución inicial, para el caso de este problema se optó porque esta solución fuera de carácter factible, por tal motivo se construyó una heurística la cual toma como principio el criterio del vecino más cercano, este último tiene por objetivo buscar el nodo más cercano al nodo actual. Para

este ejercicio la distancia no es el único factor que influye para que la solución una solución factible de calidad, ya que hay otros aspectos a tener en cuenta como lo son las ventanas de tiempo, la capacidad de los vehículos y el horizonte de planeación, pues estas repercuten en la calidad de las rutas obtenidas y la factibilidad de la solución, por tal razón se hace evidente que esta heurística adaptada tenga en cuenta estos factores adicionales y encuentre la mejor solución factible dentro de sus respectivas limitaciones de búsqueda.

Un ejemplo de solución para esta heurística teniendo en cuenta todo lo mencionado anteriormente se encuentra en el **apéndice C.**

Para la obtención de la solución inicial para la búsqueda tabú en la parte 1 se presenta el pseudocódigo de la heurística adaptada del vecino más cercano para los proveedores, el cual se puede ver en la tabla 2.

Tabla 2

Pseudocódigo del algoritmo adatado del vecino más cercano parte 1.

Algoritmo adaptado del vecino más cercano parte 1.
Input: *I*
Output: *S, C*
1: Calcular la matriz de distancias entre los nodos de los proveedores
2: Crear un vector llamado "rutas"
3: Hacer que el primer elemento de "rutas" sea el crossdock i=NO
4: ***WHILE*** *YA < N*
5: Identificar el nodo j más cercano a i
6: pruta = pruta + oferta(j)
7: ***IF*** *pruta > CMAX THEN*
8: es infactible, terminar la ruta
9: ***ELSE IF*** *pruta <= CMAX* ***THEN***
10: agregar el proveedor j a la ruta

Continuación Tabla 2

Pseudocódigo del algoritmo adatado del vecino más cercano parte 1.

11: calcular el truta y la druta
12: YA = YA + 1
13: ***END***
14: ***UNTIL*** *se cumple el criterio de parada*
15: Calcular el costo C de la solución S

El algoritmo comienza introduciendo los parámetros de la instancia que se desee evaluar *(I)*, acto seguido, este construye una matriz de distancias con base en las coordenadas cartesianas de cada uno de los nodos, para después crear un vector en donde se va a guardar el orden de visita para cada proveedor excluyendo el nodo del cross-dock el cual tiene una notación diferente *(N0).*

El funcionamiento del algoritmo se basa en identificar el nodo j más cercano al nodo i mientras comprueba simultáneamente la capacidad de los vehículos mediante la fórmula $pruta = pruta + oferta\ (j)$, acto seguido el algoritmo comprueba si se excede o no la capacidad del vehículo mediante la fórmula $pruta > CMAX$ (capacidad máxima del vehículo), en dado caso de que el peso de la ruta exceda la capacidad máxima se cancela el proveedor evaluado para no visitarlo en esa ruta y se procede a verificar la condición con el siguiente proveedor más cercano, por otro lado, si el peso de la ruta es menor o igual a la capacidad máxima se visita ese proveedor, se calcula el tiempo y la distancia de la ruta y se repite el proceso hasta que se hayan atendido a todos los proveedores. Una vez cumplido ese criterio de parada, el algoritmo imprime un vector llamado rutas, el cual corresponde a la solución inicial *(S)* para el algoritmo de búsqueda tabú para la parte 1 junto a su respectivo costo de enrutamiento *(C).*

Para la obtención de la solución inicial para la búsqueda tabú en la parte 2 se creó una variante del vecino más cercano que incluyera las ventanas de tiempo y el horizonte de planeación, el seudocódigo para esta variante se encuentra en la tabla 3.

Tabla 3

Pseudocódigo del algoritmo adatado del vecino más cercano parte 2.

Algoritmo adaptado del vecino más cercano parte 2.
Input: *I*
Output: *S, C*
1: Calcular la matriz de distancias entre los nodos de los proveedores
2: Crear un vector llamado "rutas"
3: Hacer que el primer elemento de "rutas" sea el crossdock i=NO
4: ***WHILE*** *YA < N*
5: Identificar el nodo j más cercano a i
6: pruta = pruta + demanda(j)
7: truta = truta + tiempo (i, j)
8: druta = druta + distancia (i, j)
9: ***IF*** *(truta > ventana final de j) o (pruta > CMAX)* ***THEN***
10: es infactible, terminar la ruta
*11: **ELSE IF** (truta < ventana inicial de j) y (pruta <= CMAX) **THEN***
*12: **IF** la distancia de i a j+1 es la misma de i-j **THEN***
*13: **IF** j+1 está más cerca a la ventana inicial **THEN***
14: hacer j=j+1
*15: **END***
*16: **END***
17: agregar el proveedor j a la ruta
18: truta = ventana inicial de j + tiemposervicio(j)
19: YA = YA + 1
*20: **ELSE IF** (truta >= ventana inicial de j) y (truta <= ventana final de j) y (pruta <= CMAX) **THEN***
*21: **IF** la distancia de i a j+1 es la misma de i-j **THEN***
*22: **IF** j+1 está más cerca a la ventana final **THEN***

Continuación Tabla 3

Pseudocódigo del algoritmo adatado del vecino más cercano parte 2.

23: hacer j=j+1
24: ***END***
25: ***END***
26: agregar el cliente j a la ruta
27: truta = truta + tiemposervicio(j)
28: YA = YA + 1
29: ***END***
30: ***UNTIL*** *se cumple el criterio de parada*
31: Calcular el costo C de la solución S

El algoritmo comienza introduciendo los parámetros de la instancia que se desee evaluar *(I),* acto seguido, este construye una matriz de distancias con base en las coordenadas cartesianas de cada uno de los nodos, para después crear un vector en donde se guarda el orden de visita para cada cliente excluyendo el nodo del cross-dock el cual tiene una notación diferente *(N0).*

El funcionamiento del algoritmo se basa en identificar el nodo j más cercano al nodo i mientras comprueba simultáneamente las condiciones de capacidad, tiempo (ventanas de tiempo y horizonte de planeación) y distancia de las rutas de los vehículos mediante las fórmulas $pruta = pruta + demanda\ (j)$, $truta = truta + (i.)$ y $druta = druta + distancia\ (i,j)$ respectivamente.

En caso de que el tiempo de la ruta sea mayor al tiempo final de la ventana de tiempo en j o que la demanda del cliente sea mayor a la capacidad actual del vehículo, se elimina ese cliente y se procede a evaluar el próximo más cercano. Por otro lado, si el cliente llega a ser factible cumpliendo las restricciones planteadas, este será agregado al vector solución de la ruta, luego se calcula el tiempo y distancia de esta, para posteriormente repetir el proceso hasta

que se hayan atendido a todos los clientes. Una vez cumplido ese criterio de parada, el algoritmo imprime un vector llamado rutas, el cual corresponde a la solución inicial *(S)* para el algoritmo de búsqueda tabú correspondiente a la parte 2 junto a su respectivo costo de enrutamiento *(C).*

Por otro lado, si se presenta el caso de tener dos o más clientes factibles con distancias iguales respecto al nodo actual el algoritmo toma otra ruta a seguir teniendo en cuenta los siguientes criterios de selección:

- En caso de que el tiempo de la ruta sea menor a la ventana de tiempo inicial en los destinos j y que la demanda de estos sea menor a la capacidad actual del vehículo, se escoge el cliente con la ventana de tiempo más cercana.

- En caso el tiempo de ruta sea mayor o igual a la ventana de tiempo inicial en alguno de los clientes j y menor o igual a la ventana de tiempo final, además de cumplir con las restricciones de capacidad, se deberá escoger al cliente con la ventana de tiempo final más próxima a terminar.

Una vez asignando el nuevo cliente se actualiza el tiempo de la ruta en curso por medio de la formula *truta = ventana inicial de j + tiemposervicio(j)* hasta haber atendido todos los clientes posibles para ese vehículo, cuando ya se hayan visitado todos los clientes el algoritmo imprime un vector llamado rutas, el cual corresponde a la solución inicial *(S)* para el algoritmo de búsqueda tabú correspondiente a la parte 2 junto a su respectivo costo de enrutamiento *(C).*

5.2. Metaheurística de búsqueda tabú

La presente sección tiene como objetivo explicar el funcionamiento del algoritmo búsqueda tabú, el cual opera bajo el concepto de visitar un vecindario distinto en cada una de

las iteraciones, esto se logra mediante la selección del mejor vecino evaluado. Para la creación de cada uno de los vecinos se optó por hacer uso de una estrategia de intercambio entre los nodos, la cual consiste en tomar uno de ellos e intercambiarlo con todas las demás ubicaciones posibles excluyendo el cross-dock (N0) como se puede observar en la *figura 8*. Una vez se hayan ejecutado todas las iteraciones programadas, el algoritmo imprime un vector solución con la nueva combinación de nodos el cual se presenta en la *figura 9*, donde las casillas resaltadas son los nodos que fueron intercambiados al final del ejercicio después de N iteraciones.

Figura 8.

Ejemplo de la estrategia de intercambio sacado de la instancia R1-25 para los proveedores.

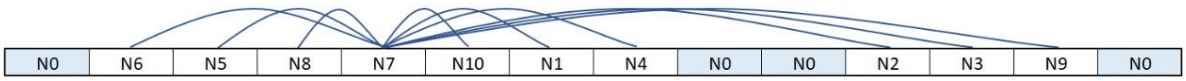
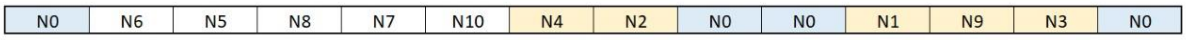


Figura 9.

Vector final de la estrategia de intercambio sacado de la instancia R1-25 para los proveedores después de 25 iteraciones.



Una vez obtenida la solución inicial mencionada en la sección anterior, dada por el algoritmo del vecino más cercano, se prosigue a la búsqueda de una mejor solución mediante la estrategia de intercambio previamente mencionada. El seudocódigo para la metaheurística de búsqueda tabú formulada para esta presente investigación se encuentra en la tabla 4.

Tabla 4

Pseudocódigo del algoritmo búsqueda tabú.

Algoritmo búsqueda tabú

1: ***Input:*** *Sa, Ca, s, TB*

2: ***Output****: S*, C**

3: Hacer S=S'=Sa*

4: ***WHILE*** *TT < TB*

5: S'=Sa

6: Generar un vector "noindices" con "no" números aleatorios entre 1 y N

7: Calcular todos los intercambios i-j y j-i de los nodos que no pertenecen a "noindices"

8: Verificar la factibilidad de cada intercambio

9: ***IF*** *el intercambio es factible* ***THEN***

10: Guardar el intercambio en la tabla de intercambios

11: ***ELSE***

12: Desechar el intercambio

13: ***END***

14: Organizar la tabla de intercambios de manera ascendente de acuerdo con el costo

15: Seleccionar el intercambio i-j de menor costo

16: ***IF*** *el intercambio i-j ya está en la lista tabú* ***THEN***

17: ***IF*** *el costo del intercambio i-j es mejor que C* o que Ca* ***THEN***

18: Se acepta el intercambio

19: ***ELSE***

20: Se desecha ese intercambio y se pasa al siguiente con menor costo

21: ***END***

22: ***END***

23: Sa=S'

24: Ca=C'

25: TT = TT + 1

26: Se agrega i-j a la lista tabú

27: ***IF*** *Sa <= S** ***THEN***

28: S=Sa*

29: C=Ca*

30: ***END***

31: ***UNTIL*** *se cumple el criterio de parada*

El algoritmo comienza leyendo los vectores de solución *(Sa)* y los costos obtenidos *(Ca)* para las partes 1 y 2, acto seguido se define un tamaño de la lista tabú *(s)* y cantidad de iteraciones globales *(TB)* y el porcentaje total de nodos a intercambiar *(no)*, para que posteriormente el algoritmo arroje un nuevo vector solución *(S*)* donde los costos *(C*)* se supone deben ser menores para las partes 1 y 2.

El algoritmo empieza por calcular un nuevo vector para cada iteración denominado *(S')* en el cual aplica la estrategia de intercambio mencionada anteriormente, una vez se hayan realizado todos los intercambios posibles con respecto al total de nodos que se desean intercambiar, el algoritmo selecciona aquellos movimientos factibles verificando cada una de las restricciones del modelo (capacidad, ventanas de tiempo y horizonte de planeación), donde luego aquellos que cumplen con el criterio de factibilidad se les saca el costo de enrutamiento y se ordenan de menor a mayor en una matriz temporal.

Posteriormente se elige aquel intercambio con menor costo y se verifica que este movimiento no se encuentre dentro de la lista tabú, de ser así, este se selecciona como el intercambio i-j de mejor costo y es aceptado. De tal manera que el vector de la solución actual se convierta en la solución inicial *(Sa = S')* junto a su respectivo costo de enrutamiento *(Ca = C')* para la siguiente iteración ($TT = TT + 1$).

Previo al inicio de la siguiente iteración, se agrega el intercambio i-j seleccionado a la lista tabú y se rectifica si la solución actual es inferior a la mejor solución encontrada ($Sa \leq S^*$), en caso de serlo se actualiza como la nueva mejor solución global (*S*=Sa y C*=Ca*).

Por último se repite el proceso hasta cumplir con el criterio de parada que corresponde a la cantidad de iteraciones especificadas al algoritmo.

En caso tal de que la mejor solución actual sea un movimiento tabú, este se somete a un criterio de aspiración, el cual consiste en que si el costo del presente intercambio es menor al mejor costo global se permitirá dicho movimiento a pesar de ser prohibido, si no es el caso, se desecha el intercambio actual, se elige el siguiente vecino con menor costo en la matriz temporal de movimientos factibles y se somete a la misma prueba en caso de estar en la lista tabú.

Una vez llegado al criterio de parada, el algoritmo imprime el vector de la mejor solución encontrada para los proveedores y clientes, junto a sus respectivos costos, incluyendo las gráficas de las rutas programadas. Un ejemplo de lo anterior se puede consultar en el **Apéndice D.**

6. Resultados obtenidos

En el presente capitulo se presenta un análisis experimental con el fin de determinar los factores más influyentes en el costo de enrutamiento a la hora de evaluar las diferentes instancias en los algoritmos previamente elaborados, seguido de esto, se exponen las tablas que contienen los resultados obtenidos con la heurística del vecino más cercano adaptado y la metaheurística búsqueda tabú con base en lo obtenido en el diseño de experimentos.

Previo a las pruebas finales de los algoritmos, estos se ensayaron con 24 problemas adaptados de la literatura basados en el repertorio de (Solomon, 2008) los cuales se adaptaron con base en la problemática propuesta en (Dondo, 2013). Las instancias mencionadas anteriormente están distribuidas en tres clases, las cuales son:

- Instancias tipo R: Se caracterizan porque la distribución geográfica de sus nodos es completamente aleatoria.
- Instancias tipo C: Se caracterizan porque los nodos están agrupados geográficamente por clústeres.

- Instancias tipo RC: Se caracterizan por ser un hibrido entre los problemas tipo R y C, es decir, algunos nodos están distribuidos aleatoriamente y otros están agrupados por clústeres.

6.1. Diseño y análisis experimental

En esta sección se realizo el analisís de comportamiento para el algoritmo búsqueda tabú, el cual permita observar la conducta del algoritmo cuando se encuentra sometido a ciertos parámetros los cuales se consideran relevantes o que pueden causar cierta influencia a la hora de obtener los resultados del algoritmo, se hizo el diseño experimental usando 3 instancias las cuales consideramos relevantes y que podrían arrojar información significativa, los problemas evaluados son RC2 de 25 nodos, junto con RC1 y R2 de 50 nodos, debido a su cantidad de nodos y la distribucion geográfica de estos. Este experimento se llevo a cabo mediante un diseño factorial de 2^3 en donde los factores a evaluar son:

- Tamaño de la lista tabú.
- Cantidad de iteraciones.
- Porcentaje de nodos intercambiados.

Los resultados individuales obtenidos para cada uno de los problemas anteriormente mencionados por medio del diseño de experimentos fueron realizados en el programa Minitab® en su versión 20.3.0.0 y se pueden consultar en el **apendice F**.

De los resultados obtenidos una vez concluido el diseño de experimentos, se deduce que entre mayor sea el tamaño de las instacias el efecto de los factores porcentaje de nodos intercambiados y cantidad de iteraciones globales toma mayor relevancia, ya que en instancias más pequeñas no es tan evidente el efecto que tienen estos factores sobre los costos de enrutamiento. Por otro lado, tambien se pudo observar que la incidencia de los efectos causados

por las interacciones entre los factores pincipales depende de la distribucion geografica del problema (R, C y RC).

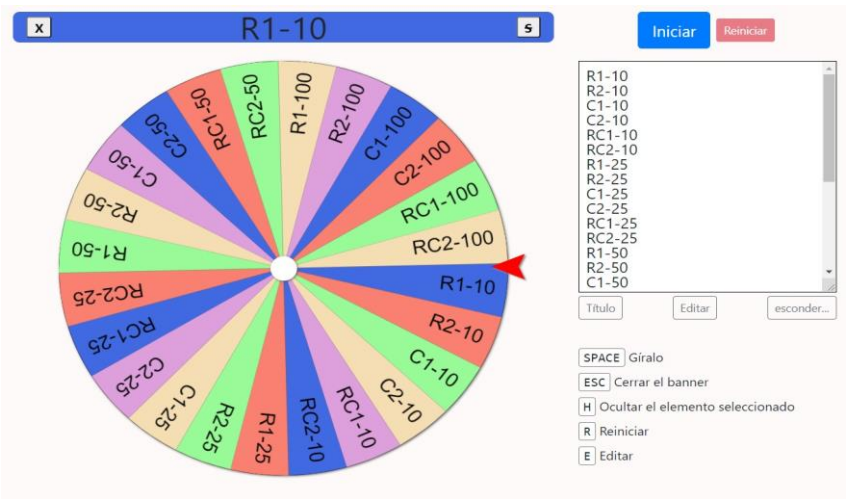
Para finalizar se puede decir que no se encontró evidencia significativa para afirmar que l tamaño de la lista tabu tenga influencia sobre los costos de enrutamiento.

6.2. Análisis de los resultados obtenidos por el algoritmo

Para la recolección de datos y análisis de resultados se escogieron aleatoriamente 10 problemas los cuales permitieron evaluar los algoritmos en toda clase de condiciones, con el fin de determinar la eficiencia del algoritmo búsqueda tabú con respecto a la solución inicial proporcionada por el algoritmo adaptado del vecino más cercano, esto debido a que se considera que con esta cantidad de ejercicios se pueden sacar buenas conclusiones en tiempos de análisis razonables para el software disponible por parte del equipo investigador. El método usado para la selección de las 10 fue mediante el uso de la ruleta aleatoria fue generado a traves de la página web Philiapp (https://es.piliapp.com/random/wheel/), la cual se muestra en la *figura 10.*

Figura 10.

Método utilizado para la selección de instancias.



La herramienta fue desarrollada en el software Matlab ®, versión 2020b y ejecutada en un computador Acer Nitro 5 AN515-53 con sistema operativo Windows 10 Home Single Languaje 64 bits, con un procesador intel® core™ i5-8300H CPU 3,95 GHz (8 CPUs) y 16 GB de memoria RAM y disco SSD de 1 TB.

Debido a que en el diseño de experimentos realizado en la subsección 6.1 se encontró que la cantidad de nodos a evaluar y el número de iteraciones totales son parámetros que pueden tener efectos significativos en los costos de enrutamiento totales del problema, mientras que con el tamaño de la lista tabú no se evidencia un efecto relevante en los resultados, se optó por modificar únicamente el número de iteraciones globales y la cantidad de nodos a intercambiar sin variar el tamaño de lista tabú en la evaluación de las vecindades para el algoritmo búsqueda tabú.

Para definir el tamaño de la lista se consultó la opinión de diversos autores, Sadiq & Youssef (1999) afirman que para algunos experimentos se suelen usar comúnmente tamaños de lista entre 5 y 12, por otro lado, Bodas (2017) afirma que para los tamaños de lista tabú se vienen utilizando valores de 7, n o $\sqrt{n}$, donde n es el número de nodos en el problema, por otro lado, en la tesis de (Gómez & Rangel, 2011) se llegó a la conclusión de que para la problemática de ruteo de vehículos con ventanas de tiempo se obtienen los mejores resultados cuando el tamaño de lista es de 7.

Con base en la información mencionada anteriormente se decidió usar una lista tabú de tamaño 7 para realizar las pruebas, el algoritmo tomó los resultados evaluando el 100% de los nodos y todos sus intercambios posibles, mientras que posteriormente estos iban disminuyendo con el fin de reducir los tiempos de cómputo a unos que se pueden considerar aceptables y darle la oportunidad al algoritmo de buscar en otros vecindarios que normalmente no tendría

en cuenta. Esta forma de evaluar los nodos permite al algoritmo en muchos casos tener mejores resultados en una menor cantidad de tiempo. Los resultados obtenidos se pueden consultar en el **apéndice E.**

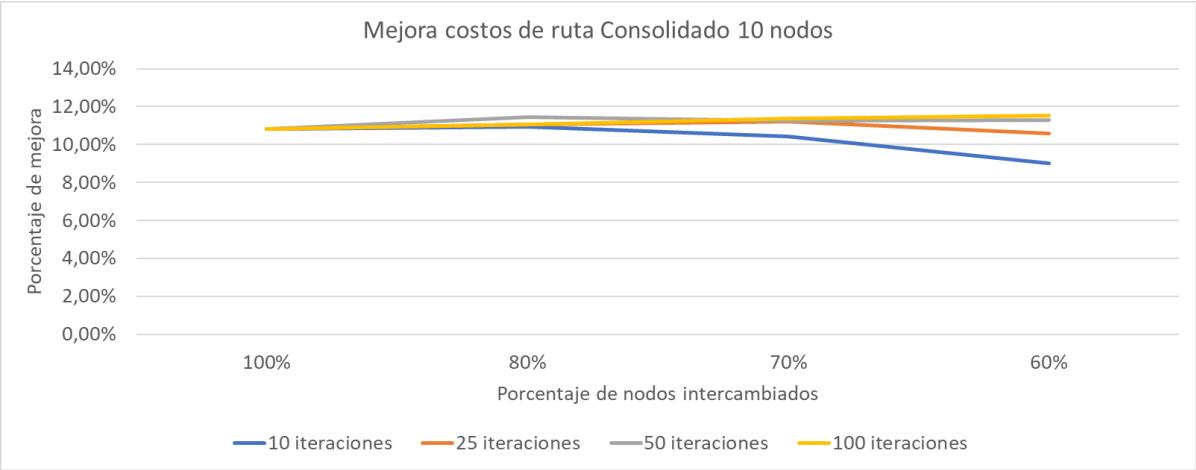
Para el análisis de los resultados y la selección de las mejores configuraciones en relación calidad de la respuesta y tiempo de cómputo, se definieron los siguientes intervalos de tiempo que se consideran razonables a criterio de los evaluadores de este proyecto, ya que es un tiempo de espera para la obtención de los resultados que se puede considerar aceptable en una aplicabilidad de la vida real para los siguientes tamaños de instancia:

- 10 nodos: Tiempo máximo 2 minutos.
- 25 nodos: Tiempo máximo 5 minutos.
- 50 nodos: Tiempo máximo 10 minutos.
- 100 nodos: Tiempo máximo 20 minutos.

De los resultados obtenidos en el apéndice D, se construyeron las gráficas que se pueden observar en las *figuras 11, 12, 13 y 14* en donde se observan los porcentajes de mejora en los costos de ruta de la búsqueda tabú con respecto al vecino más cercano, esto debido a que no se consideró relevante hacer el análisis sobre los costos totales ya que el algoritmo construido optimiza únicamente las distancias recorridas y no la cantidad de rutas programadas, además de que evaluar los costos totales no permite apreciar la mejora real en los costos, de igual manera, se puede apreciar el comportamiento promedio del algoritmo al someterse a diferentes configuraciones como lo viene a ser la cantidad de iteraciones y de nodos a intercambiar.

Figura 11.

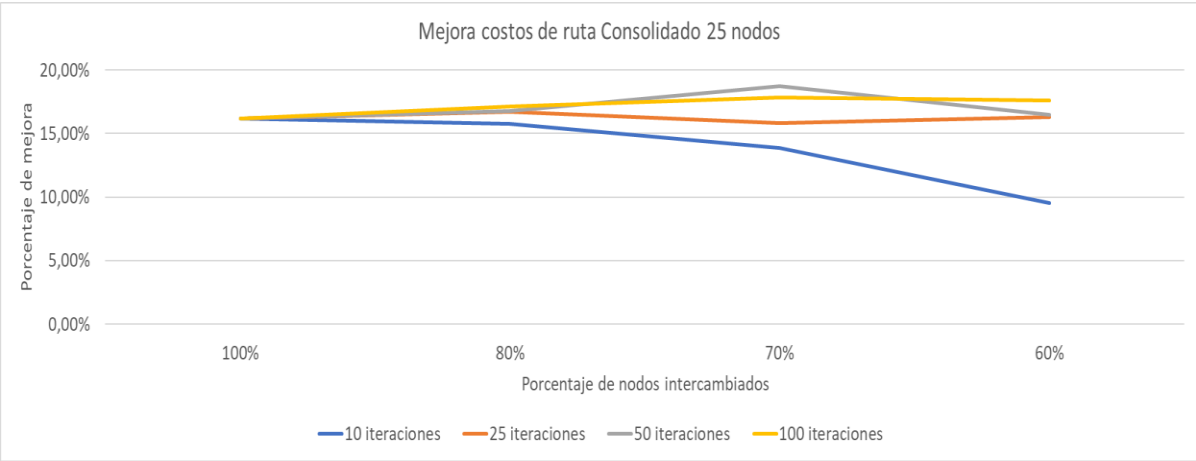
Promedios de mejora para la búsqueda tabú con problemas de 10 nodos.



De la figura 11 se puede concluir que, para instancias de 10 nodos, la mejor configuración se da en 50 iteraciones evaluando un 80% de los nodos, ya que se obtiene una mejora promedio en los costos de ruta de un 11,45% y un tiempo de cómputo estimado de 54,04 segundos, que es aproximadamente 1 minuto

Figura 12.

Promedios de mejora para la búsqueda tabú con problemas de 25 nodos.

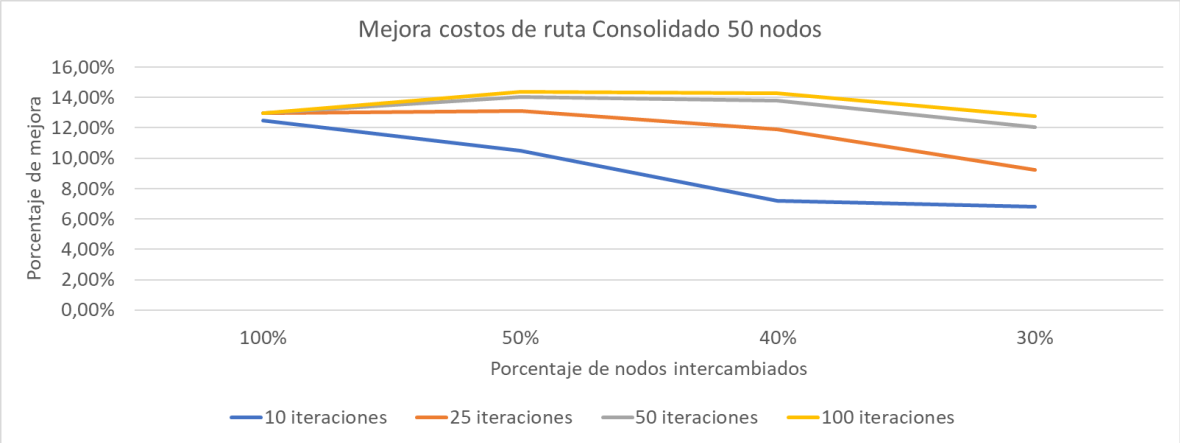


De la figura 12 se puede concluir que, para instancias de 25 nodos, la mejor configuración se da en 50 iteraciones evaluando un 70% de los nodos, ya que se obtiene una

mejora promedio en los costos de ruta de un 18,73% y un tiempo de cómputo estimado de 181,76 segundos que es aproximadamente 3 minutos

Figura 13.

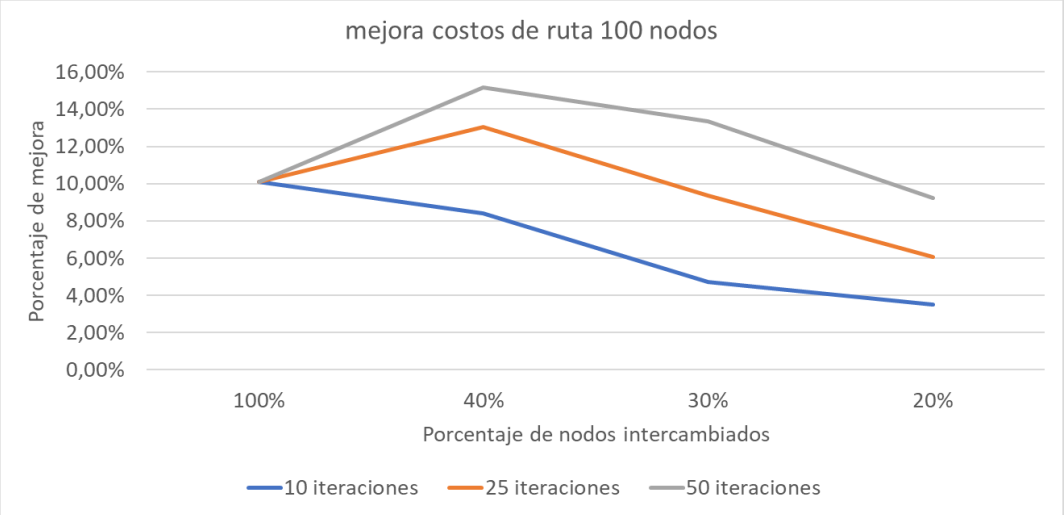
Promedios de mejora para la búsqueda tabú con problemas de 50 nodos.



De la figura 13 se puede concluir que, para instancias de 25 nodos, la mejor configuración se da en 100 iteraciones evaluando un 50% de los nodos, ya que se obtiene una mejora promedio en los costos de ruta de un 14,4%, sin embargo el tiempo de cómputo promedio para esta configuración es de 1142,21 segundos que es aproximadamente 19 minutos, lo cual se considera razonable según los criterios previamente establecidos, por tal motivo se definió que la mejor configuración son 50 iteraciones con el 50% de los datos, ya que la calidad de la respuesta varia significativamente con respecto a la primera configuración, siendo esta de un 14,03% y teniendo un tiempo de cómputo promedio de 562,56 segundos, lo cual es aproximadamente 9,37 minutos.

Figura 14.

Promedios de mejora para la búsqueda tabú con problemas de 100 nodos.



Para el caso de la instancia de 100 nodos, se optó por realizar un máximo de 50 iteraciones, ya que los tiempos fueron muy superiores a lo que se considera un intervalo aceptable, de igual manera se obtuvo que la mejor configuración en relación calidad de la respuesta y tiempo de computación se encuentra a las 25 iteraciones con un 40% de los nodos, donde se obtuvo una mejora en los costos de ruta del 13,04% con tiempo de cómputo de 697,58 segundos, que es aproximadamente 11,62 minutos, ya que con 50 iteraciones se excede el límite aceptable de tiempo.

Tabla 5

Consolidado de resultados del algoritmo.

Consolidado		valor vecino más cercano	Valor promedio de los resultados tabú	% de mejora
		10 nodos		
R1-10	Costo de rutas	$ 533,60	$ 462,63	13,30%
	Costo total	$ 1.633,60	$ 1.564,42	4,23%
C2-10	Costo de rutas	$ 637,20	$ 545,58	14,38%
	Costo total	$ 1.937,20	$ 1.845,58	4,73%
RC1-10	Costo de rutas	$ 560,90	$ 532,77	5,02%
	Costo total	$ 1.560,90	$ 1.532,77	1,80%

Continuación Tabla 5

Consolidado de resultados del algoritmo.

10 nodos promedio	**% promedio de mejora costos de rutas**				**10,90%**
	% promedio de mejora costos totales				**3,59%**
	25 nodos				
C1-25	Costo de rutas	$ 589,80	$	524,79	11,02%
	Costo total	$ 3.289,80	$	3.227,63	1,89%
RC2-25	Costo de rutas	$ 1.271,10	$	1.204,43	5,25%
	Costo total	$ 3.121,10	$	3.054,43	2,14%
R2-25	Costo de rutas	$ 944,50	$	792,71	16,07%
	Costo total	$ 2.644,50	$	2.492,71	5,74%
25 nodos promedio	**% promedio de mejora costos de rutas**				**10,78%**
	% promedio de mejora costos totales				**3,26%**
	50 nodos				
R2-50	Costo de rutas	1584,6	$	1.362,07	14,04%
	Costo total	4584,6	$	4.376,91	4,53%
RC1-50	Costo de rutas	2349,8	$	2.120,64	9,75%
	Costo total	6449,8	$	6.220,64	3,55%
C2-50	Costo de rutas	1599,2	$	1.405,66	12,10%
	Costo total	4849,2	$	4.655,66	3,99%
50 nodos promedio	**% promedio de mejora costos de rutas**				**11,97%**
	% promedio de mejora costos totales				**4,02%**
	100 nodos				
R1-100	**% promedio de mejora costos de rutas**				**9,43%**
	% promedio de mejora costos totales				**2,15%**

Finalmente, en la Tabla 5 se presenta el promedio del desempeño general para el algoritmo de búsqueda tabú en cada una de las instancias evaluadas, en donde se observa los costos obtenidos y el porcentaje de mejora con respecto al vecino más cercano.

7. Conclusiones

Con base en la revisión de la literatura se puede concluir que el problema de ruteo de vehículos (VRP) es una rama de bastante interés en el área la investigación de operaciones, debido a su gran variedad de problemáticas tratadas, las cuales son abordadas por múltiples tipos de algoritmos que van evolucionando con el tiempo, los cuales buscan sacar provecho de las herramientas computacionales de hoy en día para obtener resultados satisfactorios en

tiempos de cómputo razonables, esto se debe a que este tipo de problemas aumenta significativamente su complejidad y uso de recursos informáticos a medida que se hacen más grandes y complejos.

Por otro lado, el cross-dock es una temática que ha ido tomando mayor relevancia con el paso de los años, aumentando su popularidad en el área investigativa en países como Irán y Estados unidos, ya que es una temática bastante flexible que no se aplica exclusivamente en el campo de optimización como lo es el ruteo de vehículos, sino que se usa en otros ámbitos como lo es la logística en tareas como la planeación de inventarios y aplicación del justo a tiempo.

Dado que los problemas de VRP y sus diferentes variantes son problemas de tipo NP-hard se hace necesario el uso de algoritmos heurísticos y metaheurísticos para obtener soluciones de buena calidad en tiempos razonables.

De los resultados obtenidos en la investigación se concluye que generalmente la metaheurística de búsqueda tabú genera mejores soluciones que la respuesta obtenida por la heurística adaptada del vecino más cercano, ya que esta no se encierra en óptimos locales y puede tomar peores soluciones para llegar a una respuesta de mejor calidad. Por otro lado, si la solución inicial se genera de manera aleatoria puede darse el caso en que el algoritmo de búsqueda tabú genere soluciones de peor calidad a la obtenida por la heurística previamente mencionada, ya que el vector aleatorio podría incluso tener mayor cantidad de vehículos asignados, lo que conlleva en una desventaja notable para el algoritmo tabú propuesto en este trabajo, ya que este solo optimiza las rutas sin eliminar vehículos.

Para el algoritmo de búsqueda tabú propuesto, se concluye que resulta más efectivo no evaluar la totalidad de los intercambios posibles, sino una cantidad menor donde se escoja aleatoriamente los nodos a intercambiar, esto por múltiples razones. En primera instancia se pueden obtener mejores respuestas o de una calidad similar en tiempos de cómputo mucho más

cortos, por otro lado, al tener esta componente aleatoria se le permite al algoritmo evaluar vecindarios que usualmente descartaría, lo que puede generar mejores respuestas y por último, si la lista tabú no es lo suficientemente efectiva el algoritmo puede caer en ciclos, lo que conlleva en no mejorar la respuesta y aumentar inútilmente los tiempos de cómputo, mientras que con el componente de aleatoriedad es menos probable que esto se manifieste.

Se observa que el algoritmo propuesto resulta efectivo para instancias de tamaño reducido, ya que en instancias de 100 nodos o más los tiempos de cómputo son excesivamente altos y no resultan viables para una aplicabilidad en la vida real. De igual manera, se recomienda hacer uso de la componente de aleatoriedad en la evaluación de los nodos para encontrar un equilibrio entre la calidad de la respuesta y el tiempo de cómputo que se esté dispuesto a esperar por parte del evaluador.

Por último, aunque el diseño de experimentos concluya que el tamaño de la lista tabú y las interacciones de los efectos principales no influyen significativamente en la calidad de la respuesta obtenida, no hay evidencia suficiente para generalizar esta afirmación, esto debido a que como se pudo observar en cada una de las instancias analizadas, la influencia de los factores depende mayormente de la naturaleza del problema, como lo viene a ser la distribución geográfica de los nodos y la duración de las ventanas de tiempo, por lo que estos pueden llegar a tener efectos significativos en la respuesta obtenida para algunos problemas.

8. Recomendaciones

Con el fin de mejorar los resultados obtenidos por el algoritmo TS se propone hacer uso de otras heurísticas como lo viene a ser el algoritmo de ahorros de Clarke and Wright, los cuales permitan explorar otras soluciones iniciales para el algoritmo de búsqueda tabú.

Se recomienda hacer uso de una estrategia de tabú-inserción o similar, la cual permita eliminar rutas de hasta un solo nodo o más para que sean atendidas por otros vehículos, lo cual permitiría mejorar los costos fijos, ya que estos no pueden ser modificados con la estrategia de tabú-intercambio propuesta en esta investigación, lo que daría lugar a obtener mejores soluciones globales no optimicen únicamente los costos de ruta sino los costos totales del problema.

Se propone el uso de este trabajo como punto de partida para futuras investigaciones las cuales quieran abordar más a fondo el problema, dando como ejemplos el añadir ventanas de tiempo a los proveedores, utilizar más de un centro de cross-dock, manejar una flota de vehículos heterogénea o inclusive un repertorio de diferentes productos.

Por último, si se desea aplicar el presente algoritmo en campos donde se requiera evaluar instancias con gran número de nodos y con mayor rapidez en los tiempos de cómputo, se debe contar con la tecnología necesaria para esto, ya que en esta investigación la capacidad computacional que se disponía fue un limitante para esto.

Referencias bibliográficas

Ahkamiraad, A., & Wang, Y. (2018). Capacitated and multiple cross-docked vehicle routing problem with pickup, delivery, and time windows. *Computers and Industrial Engineering*, *119*, 76–84. https://doi.org/10.1016/j.cie.2018.03.007

Armentano, V. A., & Yamashita, D. S. (2000). Tabu search for scheduling on identical parallel machines to minimize mean tardiness. Journal of Intelligent Manufacturing, 11(5), 453–460. https://doi.org/10.1023/A:1008918229511

Bianchic, L., Birattari, M., Chiarandini, M., Manfrin, M., Mastrolilli, M., Paquete, L., Rossi-Doria, O., & Schiavinotto, T. (2004). Metaheuristics for the vehicle routing problem with stochastic demands. Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 3242(1), 450–460. https://doi.org/10.1007/978-3-540-30217-9_46

Blum, C., & Roli, A. (2003). Metaheuristics in Combinatorial Optimization: Overview and Conceptual Comparison. ACM Computing Surveys, 35(3), 268–308. https://doi.org/10.1145/937503.937505

Bodas, R. (2017). La metaheurística de Búsqueda Tabú aplicada al problema de Enrutamiento de Vehículos.

Buijs, P., Iris, V, & Carlo, H. (2014). “Synchronization in Cross-Docking Networks: A Research Classification and Framework.” European Journal of Operational Research 239 (3): 593–608.

Caro, M. (2017). Modelo de optimización de ruteo - localización de vehículos con ventanas de tiempo y estructuras cross-docking en cadena de suministro sostenible de alimentos

perecederos de dos eslabones (Tesis de maestría). Universidad Tecnológica de Bolívar, Cartagena, Colombia.

Chen, H. K., Chou, H. W., & Hsu, C. Y. (2011). The linehaul-feeder vehicle routing problem with virtual depots and time windows. Mathematical Problems in Engineering, 2011. https://doi.org/10.1155/2011/759418

Cortez, A. (2004). Teoría de la complejidad computacional y teoría de la computabilidad. Revista de Investigación De, 105(1), 102–105. http://200.62.146.19/bibvirtualdata/publicaciones/risi/N1_2004/a14.pdf%0Ahttp://sisbib.unmsm.edu.pe/bibvirtualdata/publicaciones/risi/n1_2004/a14.pdf%0Ahttp://revistasinvestigacion.unmsm.edu.pe/index.php/sistem/article/view/3216

Cruz-Chávez, M. A., & Díaz-Parra, O. (1999). Ano kamihikōki kumorizora watte : Band score. Programación Matemática y Software, 1(1). http://www.publicaciones.uaem.mx/index.php/progmat/article/view/62/36

Daza, J. M., Montoya, J. R., & Narducci, F. (2009). Resolución de problema de enrutamiento de vehículos con limitaciones de capacidad utilizando un procedimiento metaheurístico de dos fases. Eia, 1(12), 23–38.

Desrochers, M., Desrosiers, J., & Solomon, M. (1992). Desrochers1992.Pdf. In Operations research (Vol. 40, Issue 2, pp. 342-354.).

Dondo, R. (2013). A Branch-and-Price Method for the Vehicle Routing Problem with Cross-Docking and Time Windows. Iberoamerican Journal of Industrial Engineering, 5(10), 16–25. https://doi.org/10.13084/2175-8018.v05n10a02

Esmizadeh, Y., & Mellat Parast, M. (2020). Logistics and supply chain network designs: incorporating competitive priorities and disruption risk management perspectives. International Journal of Logistics Research and Applications, 0(0), 1–24. https://doi.org/10.1080/13675567.2020.1744546

Fred Glover. (1989). Tabu Search - Part I. Orsa Journal on Computing, 1(3), 190–206.

Gelareh, S., Glover, F., Guemri, O., Hanafi, S., Nduwayo, P., & Todosijević, R. (2020). A comparative study of formulations for a cross-dock door assignment problem. Omega (United Kingdom), 91. https://doi.org/10.1016/j.omega.2018.12.004

Gómez, D., & Rangel, C. (2011). Formular las Metaheurísticas Búsqueda Tabú y Recocido Simulado para la solución CVRP. Tesis. http://onlinelibrary.wiley.com/doi/10.1002/cbdv.200490137/abstract

Gonzalez Y, Teran L y Orjuela J. (2010). Diseño de un modelo para la asignación y ruteo de las bombas estacionarias desde las plantas de concreto de Holcim S.A., Zona Bogotá. Tesis de especialista en Gerencia en Logística Integral. Universidad Militar Nueva Granada, Bogotá, Colombia.

González-La Rotta, E. C., & Becerra-Fernández, M. (2017). Plataformas de intercambio con ruteo de vehículos. Una revisión del estado del arte. DYNA (Colombia), 84(200), 271–280. https://doi.org/10.15446/dyna.v84n200.60868

Hasani Goodarzi, A., Tavakkoli-Moghaddam, R., & Amini, A. (2020). A new bi-objective vehicle routing-scheduling problem with cross-docking: Mathematical model and algorithms. Computers and Industrial Engineering, 149(September), 106832. https://doi.org/10.1016/j.cie.2020.106832

Hsu, C.-I., Hung, S.-F., & Li, H.-C. (2007). Vehicle routing problem with time-windows for perishable food delivery. Journal of Food Engineering, 80(2), 465–475. https://doi.org/10.1016/j.jfoodeng.2006.05.029

Kang, K. H., & Lee, Y. H. (2007). Heuristic for vehicle routing problem with perishable product delivery. Journal of Korean Institute of Industrial Engineers, 33 (2), 265–272.

Küçükoğlu, İ., & Öztürk, N. (2019). A hybrid meta-heuristic algorithm for vehicle routing and packing problem with cross-docking. Journal of Intelligent Manufacturing, 30(8), 2927–2943. https://doi.org/10.1007/s10845-015-1156-z

Ladier, A.-L., & Alpan, G. (2018). Crossdock truck scheduling with time windows: earliness, tardiness and storage policies. Journal of Intelligent Manufacturing, 29(3), 569–583. https://doi.org/10.1007/s10845-014-1014-4

Landrieu, A., Mati, Y., & Binder, Z. (2001). A tabu search heuristic for the single vehicle pickup and delivery problem with time windows. Journal of Intelligent Manufacturing, 12(5–6), 497–508.

Lee, Y. H., Jung, J. W., & Lee, K. M. (2006). Vehicle routing scheduling for cross-docking in the supply chain. Computers and Industrial Engineering, 51(2), 247–256. https://doi.org/10.1016/j.cie.2006.02.006

Lozada A, Cadena R y Díaz L. 2012. Solución del Problema de Ruteo de Vehículos con ventanas de tiempo. Tesis de Ingeniería Industrial, Universidad Industrial de Santander, Bucaramanga, Colombia.

Maldonado, C., & Gómez, N. (2010). El mundo de las ciencias de la complejidad Un estado del arte. 76. https://repository.urosario.edu.co/handle/10336/3301

Moghadam, S. S., Ghomi, S. M. T. F., & Karimi, B. (2014). Vehicle routing scheduling problem with cross docking and split deliveries. Computers and Chemical Engineering, 69, 98–107. https://doi.org/10.1016/j.compchemeng.2014.06.015

Mohtashami, A. (2015). Scheduling trucks in cross docking systems with temporary storage and repetitive pattern for shipping trucks. Applied Soft Computing Journal, 36, 468–486. https://doi.org/10.1016/j.asoc.2015.07.021

Montes E, Mora R, Silva S, Rincón E, Gutiérrez M, Velázquez P. 2019 Metaheurísticas para resolver el problema de ruteo de vehículos con ventanas de tiempo. Revista de matemática, teoría y aplicaciones 2020 27(2): 305-332

Montoya-Torres, J. R., López Franco, J., Nieto Isaza, S., Felizzola Jiménez, H., & Herazo-Padilla, N. (2015). A literature review on the vehicle routing problem with multiple depots. Computers and Industrial Engineering, 79, 115–129. https://doi.org/10.1016/j.cie.2014.10.029

Morais, V. W. C., Mateus, G. R., & Noronha, T. F. (2014). Iterated local search heuristics for the Vehicle Routing Problem with Cross-Docking. Expert Systems with Applications, 41(16), 7495–7506. https://doi.org/10.1016/j.eswa.2014.06.010

Nagy, G., & Salhi, S. (2006). Location-routing: Issues, models and methods. European Journal of Operational Research, 177(2), 649–672. https://doi.org/10.1016/j.ejor.2006.04.004

Nosrati, M., & Arshadi Khamseh, A. (2020). Distance discount in the green vehicle routing problem offered by external carriers. SN Applied Sciences, 2(8). https://doi.org/10.1007/s42452-020-03245-5

Ozden, G., & Saricicek, I. (2019). Scheduling trucks in a multi-door cross-docking system with time windows. Bulletin of the Polish Academy of Sciences: Technical Sciences, 67(2), 349–362. https://doi.org/10.24425/bpas.2019.128113

Pérez, E., & Guerrero, W. J. (2015). Optimization methods for the inventory routing problem with hard time windows. Revista Ingeniería Industrial, 14(3), 31–49.

Rashidi Komijan, A., & Delavari, D. (2017). Vehicle Routing and Scheduling Problem for a multi-period, multi-perishable product system with time window: A Case study. International Journal of Production Management and Engineering, 5(2), 45. https://doi.org/10.4995/ijpme.2017.5960

Restrepo, J., Medina, P., & Cruz, E. (2008). UN PROBLEMA LOGISTICO DE PROGRAMACION DE VEHICULOS CON VENTANAS DE TIEMPO(VRPTW) A logistic case of programming vehicle routing problem with time windows. Scientia et Technica, XIV(39), 229–234.

Rocha, L.; González, C. y Orjuela, J. (2011). Una revisión al estado del arte del problema de ruteo de vehículos: Evolución histórica y métodos de solución. En: Ingeniería, Vol. 16, No. 2, pág. 35 - 55

Romanycia, M. H. J., & Pelletier, F. J. (1985). What is a heuristic? Computational Intelligence, 1(1), 47–58. https://doi.org/10.1111/j.1467-8640.1985.tb00058.x

Rosenfeld, R., & Irazábal, J. (2020). Computabilidad, complejidad computacional y verificación de programas. In Computabilidad, complejidad computacional y verificación de programas. https://doi.org/10.35537/10915/27887

Sadiq, S., & Youssef, H. (1999). Iterative Computer Algorithms with Applications in Engineering. IEEE Computer Society Press, Los Alamitos, CA, USA. ISBN: 978-0-769-50100-0

Schweickardt, Gustavo, & Miranda, Vladimiro. (2010). Metaheurística FEPSO aplicada a problemas de Optimización Combinatoria: Balance de Fases en Sistemas de Distribución Eléctrica. Ciencia, Docencia y Tecnología, XXI(40),133-163.[fecha de Consulta 2 de Enero de 2021]. ISSN: 0327-5566. Disponible en: https://www.redalyc.org/articulo.oa?id=145/14515290006

Schwind, G. (1995). Considerations for Cross Docking. Material Handling Engineering, 50(12), 47–49.

Solomon, M. M. (2008). Solomon benchmark, recurso disponible gratuitamente en: https://www.sintef.no/projectweb/top/vrptw/solomon-benchmark/

Tarantilis, C., & Kiranoudis, C. (2001). A meta-heuristic algorithm for the efficient distribution of perishable foods. Journal of Food Engineering, 50(1), 1–9. https:// doi.org/10.1016/S0260-8774(00)00187-4

Utama, D. M., Dewi, S. K., Wahid, A., & Santoso, I. (2020). The vehicle routing problem for perishable goods: A systematic review. Cogent Engineering, 7(1). https://doi.org/10.1080/23311916.2020.1816148

Vahdani, B., Reza, T. M., Zandieh, M., & Razmi, J. (2012). Vehicle routing scheduling using an enhanced hybrid optimization approach. Journal of Intelligent Manufacturing, 23(3), 759–774. https://doi.org/10.1007/s10845-010-0427-y

Van, J., Valckenaers, P., Cattrysse D,. (2012). Cross-docking: State of the art. Omega, 40–46.

Wang, J., Ranganathan Jagannathan, A. K., Zuo, X., & Murray, C. C. (2017). Two-layer simulated annealing and tabu search heuristics for a vehicle routing problem with cross docks and split deliveries. Computers and Industrial Engineering, 112, 84–98. https://doi.org/10.1016/j.cie.2017.07.031

Wen, M., Larsen, J., Clausen, J., Cordeau, J. F., & Laporte, G. (2009). Vehicle routing with cross-docking. Journal of the Operational Research Society, 60(12), 1708–1718. https://doi.org/10.1057/jors.2008.108

Yazdani, M., Naderi, B., Rahmani, S., & Rahmani, S. (2017). Truck routing and scheduling for cross-docking in the supply chain: Model and solution method. RAIRO - Operations Research, 51(3), 833–856. https://doi.org/10.1051/ro/2016067

Yin, P. Y., & Chuang, Y. L. (2016). Adaptive memory artificial bee colony algorithm for green vehicle routing with cross-docking. Applied Mathematical Modelling, 40(21–22), 9302–9315. https://doi.org/10.1016/j.apm.2016.06.013