Capítulo 1

Exploración de vecindades del VRP utilizando modelos de optimización continua

En este capítulo se presentan tres modelos de optimización lineal para determinar la mejor solución en las vecindades de reinserción de un cliente, reinserción de subrutas e intercambio de subrutas para el Problema de Enrutamiento de Vehículos con restricción de Capacidad.

Aunque existen diferentes estrategias para realizar la exploración de una vecindad como la búsqueda aleatoria o búsqueda hasta la primera mejora, la única que garantiza la obtención del óptimo de la vecindad es la búsqueda exhaustiva. Sin embargo, al modelar la exploración como un problema de optimización, se pudiera realizar una búsqueda exhaustiva de la vecindad sin tener que construir todas las soluciones vecinas permitiendo encontrar el óptimo en menos tiempo.

Además, gracias a los avances en algoritmos de optimización [1], los modelos de optimización son una herramienta que permiten resolver problemas complejos con una mayor rapidez computacional en comparación a una búsqueda exhaustiva.

La estructura de este capítulo es la siguiente. En la sección 1.1 se presentan algunas características del VRP, que permiten definir las variables, parámetros y los modelos de optimización. En la sección 1.2, se presenta un modelo de optimización para la vecindad de reinserción de clientes. Este modelo, por su simplicidad, sirve como punto de partida para los modelos más avanzados que se presentan en las secciones 1.3 y 1.4.

1.1. Características del VRP

Una de las características del Problema de Enrutamiento de Vehículos es que sus soluciones se pueden representar como un grafo dirigido y ponderado. Sea $(G = V, E)$ el grafo dirigido y ponderado que representa una solución del VRP, donde cada vértice $v \in V$ representa a un cliente del sistema, y el vértice especial v_0 corresponde al depósito central. Este nodo actúa como punto de inicio y fin de cada ruta de servicio. Los arcos se expresan mediante pares ordenados de nodos $j = \langle j_s, j_e \rangle$, donde j_s es el nodo de inicio y j_e el nodo final del arco j. Cada arco posee un costo que representa la distancia entre el nodo inicio y el nodo final.

Para ilustrar esta representación, se muestra una posible solución del CVRP con dos rutas y cinco clientes (ver Figura 1.1), y su representación como un grafo (Figura 1.2). Los números encima de los arcos representan la distancia entre los clientes del arco.

$$\begin{pmatrix} r1 : (1, 2, 3) \\ r2 : (4, 5) \end{pmatrix}$$

Figura 1.1: Solución para un CVRP con 5 clientes y 2 vehículos.

$$0 \xrightarrow{1} 1 \xrightarrow{9} 2 \xrightarrow{5} 3 \xrightarrow{3} 0$$
$$0 \xrightarrow{7} 4 \xrightarrow{2} 5 \xrightarrow{1} 0$$

Figura 1.2: Representación como grafo.

Otra característica que se debe tener en cuenta para la exploración de vecindades son las subrutas de una solución. Las subrutas representan segmentos consecutivos de clientes dentro de una ruta que se pueden intercambiar o reinsertar para construir nuevos vecinos.

Formalmente, una subruta se define como una secuencia continua de clientes dentro de una misma ruta. En la figura 1.3 de la página 3, se muestran ejemplos de subrutas para la solución de la figura 1.2.

Las subrutas están delimitadas por dos arcos, j_1 y j_2. Cuando se interpreta que las subrutas están delimitadas por los arcos j_1 y j_2, la subruta comienza en el cliente siguiente a j_1 y termina en el cliente anterior a j_2. Por ejemplo, en el grafo que se representa en la figura 1.2, es posible identificar subrutas como las que se muestran en la figura 1.3 de la página 3.

Una vez definido cómo representar las soluciones y cómo se componen de rutas y subrutas, el siguiente paso para la creación de los modelos de optimización es describir cómo explorar las vecindades de estas soluciones.

Subruta de la ruta 1: $[1 \rightarrow 2]$
Subruta de la ruta 2: $[4]$
Subruta de la ruta 1: $[1 \rightarrow 2 \rightarrow 3]$

Figura 1.3: Ejemplos de subrutas del grafo de la Figura 1.2

1.1.1. Exploración de una vecindad

La exploración de una vecindad forma parte del proceso de búsqueda de mejores soluciones. La idea es generar soluciones cercanas a una dada, con el objetivo de encontrar vecinos que mejoren la solución actual.

Una estrategia común para explorar vecindades consiste en extraer un cliente de su posición original y reinsertarlo en distintas ubicaciones dentro de la solución. Este proceso implica dos operaciones: la extracción, que consiste en eliminar temporalmente un nodo de su posición en una ruta, y la inserción, que reubica dicho nodo en una nueva posición, ya sea en la misma ruta o en una diferente.

Además de permitir la exploración de la vecindad de reinserción de un cliente, las operaciones de extracción e inserción pertenecen a otros movimientos más complejos. Por ejemplo, al aplicar la extracción e inserción de manera simultánea a dos clientes, se pueden explorar vecindades de intercambio de clientes. De forma similar, si en lugar de un solo nodo se extrae y se reubica un conjunto consecutivo de clientes, se modela la vecindad de reinserción de subrutas.

Al generar estas soluciones en problemas como el CVRP, es posible que algunas soluciones violen las restricciones de capacidad de este tipo de problemas. Para evitar que las soluciones que no cumplan las restricciones del problema, existen dos enfoques, el rechazo explicito, y la penalización en la función de costo. En **el rechazo explícito**, las soluciones que incumplen las restricciones se descartan inmediatamente. Mientras que en **la penalización en la función de costo**, se aceptan soluciones que no cumplen las restricciones, incorporando un término de penalización. En este trabajo se asume que la exploración se realiza penalizando soluciones que no son factibles.

Las operaciones de extracción e inserción pueden formalizarse matemáticamente mediante variables binarias, facilitando su integración en modelos de programación entera. Al formalizar la extracción e inserción, se puede realizar la exploración de vecindades utilizando modelos de optimización. En las siguientes secciones, se detallarán estos modelos de optimización.

1.2. Modelo de optimización para la vecindad de reinserción de un cliente

Al explorar la vecindad de reinserción de un cliente, es necesario tomar dos decisiones: qué cliente seleccionar y dónde insertarlo. Además, se asume que en cada movimiento solo se reubica un cliente dentro de la solución. Dado que el número de rutas y la cantidad de clientes en cada una de ellas son datos conocidos, es posible formular este proceso como un problema de optimización. En este modelo, las variables de decisión representarán tanto la selección del cliente a extraer como su nueva ubicación dentro de la solución.

En esta sección se presenta un modelo de optimización lineal que permite explorar la vecindad de reinserción de un cliente. Este modelo muestra cómo se pueden utilizar modelos de optimización para evaluar modificaciones en una solución. Su propósito es representar el proceso de cambio de posición de un cliente dentro de una solución dada, con el objetivo de identificar una solución vecina que minimice el costo total. En secciones posteriores, se introducirán modelos para la exploración de vecindades con un mayor número de vecinos.

A continuación se describen las variables del modelo de optimización.

1.2.1. Variables

Las variables del modelo se denotan como $x_{r_1 i r_2 j}$, donde, i es el i-ésimo cliente de la ruta r_1 y j es la posición en la que se va a insertar dicho cliente en la ruta r_2. Las variables son variables binarias que toman los valores:

$$x_{r_1 i r_2 j} = \begin{cases} 1, & \text{si se reinserta el } i\text{-ésimo cliente de la ruta } r_1 \text{ en la posición } j \text{ de la ruta } r_2 \\ 0, & \text{en otro caso.} \end{cases}.$$

En la figura 1.4 se muestra el cambio que se realiza en la vecindad cuando la variable $x_{r_1 i r_2 j} = 1$. A la izquierda se muestra una solución con 10 clientes y 2 rutas y a la derecha el cambio que se realiza cuando $x_{2,1,1,1} = 1$, la cual indica que, el primer cliente de la ruta 2 ([2]) se inserta después del primer cliente de la ruta 1 (figura 1.4).

Para evaluar el costo de cada operación de reinserción, se necesita definir los parámetros asociados a los costos de las operaciones descritas por las variables y los correspondientes a la restricción de capacidad. A continuación se detallan estos parámetros.

$r_1 : 0 \to 1 \to 4 \to 5 \to 7 \to 9 \to 10 \to 0$
$r_2 : 0 \to 2 \to 3 \to 8 \to 6 \to 0$

$r_1 : 0 \to 1 \to 2 \to 4 \to 5 \to 7 \to 9 \to 10 \to 0$
$r_2 : 0 \to 3 \to 8 \to 6 \to 0$

Figura 1.4: Reubicación de un cliente. En rojo se resalta el cliente que cambia de posición.

1.2.2. Parámetros

Para crear el modelo de optimización, se definen los parámetros que intervienen en la evaluación de cada posible movimiento dentro de la vecindad. Estos parámetros calculan los costos asociados a extraer e insertar clientes en distintas posiciones dentro de la solución.

- *capacity*: Representa la capacidad máxima de cada vehículo. La demanda total de cualquier ruta no puede exceder este límite.
- d_{ri}: Este parámetro representa la demanda del i-ésimo cliente de la ruta r.
- $RouteDemand_r$: Define la demanda total de la ruta r en una solución dada. Se obtiene mediante:

$$RouteDemand_r = \sum_{i \in r} d_{ri}.$$

- c_{rj}: Indica el costo del arco j en la ruta r. El arco j es el arco a continuación del j-ésimo cliente de la ruta. En la figura 1.5 se muestra la posición de cada arco en una ruta.

$$r_1 : 0 \xrightarrow{0} 1 \xrightarrow{1} 2 \xrightarrow{2} 3 \xrightarrow{3} 0$$
$$r_2 : 0 \xrightarrow{0} 4 \xrightarrow{1} 5 \xrightarrow{2} 0$$

Figura 1.5: Posiciones de los arcos en una solución del VRP. El número encima del arco indica su posición en la ruta.

- P_r: Representa la suma de los costos de todos los arcos de la ruta r, lo que equivale al costo total de la ruta r para una solución dada. Este se calcula como:

$$P_r = \sum c_{rj}.$$

- $K_{r_1 i r_2 j}$: Representa la distancia entre el cliente i de la ruta r_1 y el cliente j de la ruta r_2.

  En la figura 1.6 se muestra un ejemplo de como se usa el parámetro K en la modificación de una solución. Si se extrae el cliente 2, $K_{1,1,1,3}$ representa la unión entre el primer y tercer cliente de la ruta 1 (figura 1.6 variante (b)). Si se desea insertar el cliente 2 entre los clientes 4 y 5, el parámetro $K_{1,2,2,2}$ representa la unión entre el cliente 2 y 5, mientras que $K_{2,1,1,2}$, la conexión entre 4 y 2 (figura 1.6 variante (c)).

$r_1 : 0 \xrightarrow{5} 1 \xrightarrow{8} 2 \xrightarrow{3} 3 \xrightarrow{6} 0$
$r_2 : 0 \xrightarrow{1} 4 \xrightarrow{2} 5 \xrightarrow{9} 0$

(a).

$r_1 : 0 \xrightarrow{5} 1 \xrightarrow{1} 3 \xrightarrow{6} 0$
$r_2 : 0 \xrightarrow{1} 4 \xrightarrow{1} 2 \xrightarrow{1} 5 \xrightarrow{9} 0$

(b).

$r_1 : 0 \xrightarrow{5} 1 \xrightarrow{1} 3 \xrightarrow{6} 0$
$r_2 : 0 \xrightarrow{1} 4 \xrightarrow{1} 2 \xrightarrow{1} 5 \xrightarrow{9} 0$

(c).

Figura 1.6: Función del parámetro K. En rojo los el cliente que se extrae y en azul las aristas que se forman.

Con las variables y parámetros definidos, se pueden crear las restricciones y la función objetivo del modelo de optimización. A continuación se detallan las restricciones del modelo de optimización.

1.2.3. Restricciones

El modelo posee una restricción para asegurar que las soluciones que se obtienen son válidas. La primera restricción se define a continuación:

$$\sum x_{r_1 i r_2 j} = 1.$$

Con esta restricción, como las variables del modelo de optimización son binarias, se le impone al modelo de optimización seleccionar solo una. Con esta restricción se garantiza que se selecciona un único cliente y que se inserta en un único lugar.

Para asegurar que se respeta la restricción de capacidad, a la función objetivo se le añade un factor de penalización denominado *penalty*. Este factor de penalización solo se aplica cuando se selecciona una variable y el resultado de aplicar

la operación que describe esa variable genera una solución infactible. El factor de penalización se calcula como:

$$penalty = \begin{cases} x_{r_1 i r_2 j} \cdot \alpha \cdot (RouteDemand_{r_2} + d_{r_1 i} - capacity) & (1) \\ x_{r_1 i r_2 j} \cdot \alpha \cdot (RouteDemand_{r_1} - d_{r_1 i} - capacity) & (2) \\ 0 & (3) \end{cases}$$

A continuación, se explican los casos del cálculo de *penalty* para la exploración de la vecindad de reinserción de un cliente.

1. Si al insertar el i-ésimo cliente de la ruta r_1 en la posición j de la ruta r_2, la solución es infactible.

2. Si al extraer el i-ésimo cliente de la ruta r_1, la solución es infactible.

3. Si la solución es factible.

El valor de α es una constante que indica cuanto se debe penalizar una solución en caso de ser infactible. Este factor puede ajustarse para priorizar la factibilidad sobre el costo total, dependiendo de los objetivos específicos del modelo. En este trabajo se utiliza $\alpha = 10\ 000$.

El factor de penalización se puede precalcular para cada vecino a partir de la solución inicial.

En el siguiente epígrafe, se detalla la función objetivo y el uso del factor *penalty* en la misma.

1.2.4. Función objetivo

Para realizar el cálculo de la función objetivo se crean dos términos: *Eliminar* y *Sumar* que modelan la operación de extracción y la de inserción de clientes. Para el criterio de reinserción de un cliente se calculan de la siguiente manera:

- $Eliminar_{r_1 i}$: Este término representa el costo de extraer el cliente i de su posición actual en la ruta r_1, definiendo así el nuevo costo de la ruta r_1. Se calcula como la eliminación de los arcos que conectan al cliente i, para añadir el arco entre el cliente anterior y el posterior al cliente eliminado. Matemáticamente se puede describir como:

$$Eliminar_{r_1 i} = P_{r_1} - c_{r_1 i-1} - c_{r_1 i} + K_{r_1 i-1 r_1 (i+1)}.$$

- $Sumar_{r_1 i r_2 j}$: Este término representa el costo de agregar el cliente i en la posición j dentro de la ruta r_2, otorgándole un nuevo valor al costo de esa ruta. Esta operación se calcula como la eliminación del arco j y conectando el cliente i a los nodos del arco:

$$Sumar_{r_1 i r_2 j} = P_{r_2} - c_{r_2 j} + K_{r_2 j r_1 i} + K_{r_1 i r_2 (j+1)}.$$

Una vez presentados los términos $Sumar_{r_1 i r_2 j}$ y $Eliminar_{r_1 i}$, y el factor de penalidad *penalty*, la función objetivo se define como:

$$\text{mín} \sum x_{r_1 i r_2 j} \cdot (Sumar_{r_1 i r_2 j} + Eliminar_{r_1 i} + penalty).$$

A continuación se estructura el modelo de optimización con las restricciones y la función objetivo.

1.2.5. Modelo

El modelo de optimización se estructura como:

$$\text{mín} \sum x_{r_1 i r_2 j} \cdot (Sumar_{r_1 i r_2 j} + Eliminar_{r_1 i} + penalty).$$

Sujeto a:

$$\sum x_{r_1 i r_2 j} = 1.$$

El modelo que se presenta en esta sección permite realizar una exploración de la vecindad de reinserción de un solo cliente. Sin embargo, existen vecindades con un mayor número de vecinos, como la vecindad de reinserción de subrutas. En la siguiente sección se presenta un modelo de optimización para explorar la vecindad de reinserción de una subruta.

1.3. Modelo de optimización para la vecindad de reinserción de una subruta

Para la exploración de la vecindad de reinserción de una subruta se deben tomar dos decisiones: qué subruta seleccionar y en qué posición insertarla. Estas decisiones implican determinar la ruta de origen, los clientes que conforman la subruta y la nueva ubicación donde se insertará, que puede estar en la misma ruta o en una diferente.

En la sección 1.2 se presentó un modelo de optimización para la exploración de la vecindad de reinserción de un cliente. En esta sección se introduce un modelo de optimización para explorar la vecindad de reinserción de subrutas. En este modelo, las variables de decisión representarán tanto la selección de la subruta a extraer como su nueva ubicación dentro de la solución.

A continuación, se detallan las variables del modelo de optimización.

1.3.1. Variables del modelo

Para la exploración de esta vecindad se utilizan variables $X_{r1j1j2r2i}$, donde r_1 representa la ruta de la que se extrae la subruta y r_2 representa la ruta en la que se inserta, j_1 y j_2 indican los arcos donde empieza y termina la subruta, i es la posición en r_2 donde se insertará la subruta que se extrae de r_1. Las variables toman los siguientes valores:

$$X_{r_1 j_1 j_2 r_2 i} = \begin{cases} 1, & \text{si se selecciona una subruta que comienza en el cliente } j_1 + 1 \\ & \text{y termina en el cliente } j_2 \text{ de la ruta } r_1 \text{ y se reubica en la ruta} \\ & r_2 \text{ entre los clientes } i \text{ e } i+1, \\ 0, & \text{en otro caso.} \end{cases}$$

Es importante destacar que, cuando la subruta se reubica dentro de la misma ruta ($r_1 = r_2$), deben cumplirse ciertas restricciones para asegurar la validez de la operación.

La restricción que se debe cumplir consiste en que la posición de inserción i no puede estar dentro de la subruta definida por j_1 y j_2, por lo tanto i debe cumplir una de las condiciones siguientes $i < j_1$, o $i > j_2$.

Para ilustrar un ejemplo de cómo se modifica una solución existente utilizando las variables $X_{r_1 j_1 j_2 r_2 i}$, se muestra la figura 1.7. A la izquierda se muestra una solución con 10 clientes y 2 rutas, y a la derecha el cambio que se realiza cuando $X_{1,1,3,2,2} = 1$,la cual representa que la subruta comprendida por el segundo y tercer cliente de la ruta 1 ($[4, 5]$) se reubica en la ruta 2, justo después del cliente en la segunda posición (figura 1.7).

En el siguiente epígrafe se presentan los parámetros utilizados en el modelo de optimización. Algunos de estos parámetros ya fueron descritos en la sección 1.2, mientras que otros se introducen para modelar el costo asociado a la extracción e inserción de subrutas.

$r_1 : 0 \to 1 \to 4 \to 5 \to 7 \to 9 \to 10 \to 0$ $\quad$ $r_1 : 0 \to 1 \to 7 \to 9 \to 10 \to 0$
$r_2 : 0 \to 2 \to 3 \to 8 \to 6 \to 0$ $\quad$ $r_2 : 0 \to 2 \to 3 \to 4 \to 5 \to 8 \to 6 \to 0$

Figura 1.7: Reubicación de una subruta. Se resalta en color rojo la subruta que cambia de posición.

1.3.2. Parámetros

A continuación, se enumeran los parámetros previamente definidos:

- *capacity*: Representa la capacidad máxima de cada vehículo.
- d_{ri}: Este parámetro representa la demanda del i-ésimo cliente de la ruta r.
- $RouteDemand_r$: Define la demanda total de la ruta r en una solución dada.
- c_{rj}: Indica el costo del arco j en la ruta r.
- P_r: Representa el costo total de la ruta r para una solución dada.
- $K_{r_1 i r_2 j}$: Representa la distancia entre el cliente i de la ruta r_1 y el cliente j de la ruta r_2.

Los parámetros a continuación se introducen para calcular el costo las subrutas:

1. $S_{rj_1j_2}$: Define la suma de los costos de los arcos de la subruta comprendida entre los clientes $j_1 + 1$ y j_2. Representa la suma de todos los arcos de la subruta y se calcula mediante:

$$S_{rj_1j_2} = \sum_{i=j_1+1}^{j_2-1} c_{ri}.$$

   - Si la subruta contiene solo un cliente, este parámetro es igual a 0.
   - Si la subruta incluye todos los clientes de una ruta, el valor de $S_{rj_1j_2} = P_r$ menos la distancia del depósito al primer cliente de la ruta y la distancia del último cliente de la ruta al depósito.

2. $D_{rj_1j_2}$: Representa la demanda total de la subruta desde el cliente en la posición $j_1 + 1$ hasta el cliente en la posición j_2 de la ruta r. Se define como:

$$D_{rj_1j_2} = \sum_{i=j_1+1}^{j_2-1} d_{ri}.$$

En la figura 1.8 se muestra una solución con 5 clientes y 2 rutas con el objetivo de ilustrar el valor de estos parámetros. Los números encima de cada arco es el costo del arco, mientras que los números encima de cada cliente representan la demanda de esos clientes.

$r_1 : 0 \xrightarrow{2} 1 \xrightarrow{6} 2 \xrightarrow{2} 3 \xrightarrow{7} 0$ $\qquad S_{1,0,3} = 8$

$r_2 : 0 \xrightarrow{1} 4 \xrightarrow{4} 5 \xrightarrow{5} 0$ $\qquad S_{2,0,2} = 4$

$r_1 : 0 \xrightarrow{4} 1 \xrightarrow{3} 2 \xrightarrow{1} 3 \rightarrow 0$ $\qquad D_{1,0,3} = 8$

$r_2 : 0 \xrightarrow{2} 4 \xrightarrow{3} 5 \rightarrow 0$ $\qquad D_{2,0,2} = 5$

Figura 1.8: Ejemplo del valor de los parámetros $S_{rj_1j_2}$ y $D_{rj_1j_2}$. El valor de cada parámetro corresponde a la subruta con el mismo color.

Los parámetros descritos en esta sección permiten el cálculo de los costos de cada subrutas de la solución. En el próximo epígrafe se muestra como calcular el factor de penalización utilizando estos parámetros.

1.3.3. Restricciones

Al igual que el modelo que se presenta en la sección 1.2, el modelo de optimización que se estructura en esta sección posee una restricción para la selección de una variable e incluye el factor de penalización *penalty* en la función objetivo. La restricción para la selección de una variable se define como:

$$\sum X_{r_1j_1j_2r_2i} = 1.$$

La forma de calcular el factor *penalty* para la vecindad de reinserción de subrutas es la siguiente:

$$penalty = \begin{cases} X_{r_1j_1j_2r_2i} \cdot \alpha \cdot (RouteDemand_{r_2} + D_{r_1j_1j_2} - capacity) & (1) \\ X_{r_1j_1j_2r_2i} \cdot \alpha \cdot (RouteDemand_{r_1} - D_{r_1j_1j_2} - capacity) & (2) \\ 0 & (3) \end{cases}$$

A continuación, se explican los casos del cálculo de *penalty* para la exploración de la vecindad de reinserción de subrutas.

1. Si al insertar la subruta, la solución es infactible.
2. Si al eliminar la subruta, la solución continúa siendo infactible.
3. Si la solución es factible.

En el siguiente epígrafe, se muestra como se incluye el factor de penalización en el cálculo de la función objetivo.

1.3.4. Función Objetivo

Para el cálculo de la función objetivo en la vecindad de reinserción de subrutas, se emplean los términos $Sumar_{r_1 j_1 j_2 r_2 i}$ y $Eliminar_{r_1 j_1 j_2}$, siguiendo el mismo enfoque utilizado en la vecindad de reinserción de un cliente (sección 1.2). Los términos $Sumar_{r_1 j_1 j_2 r_2 i}$ y $Eliminar_{r_1 j_1 j_2}$ para la vecindad de reinserción de subrutas se calculan como:

- $Eliminar_{r_1 j_1 j_2}$: Este término representa el costo de eliminar una subruta específica, definiendo así el nuevo costo de la ruta r_1. Se calcula como la eliminación del peso de la subruta (el parámetro $S_{r_1 j_1 j_2}$) y de los arcos en los extremos de la misma, para añadir el arco entre los clientes que se desconectan al extraer la subruta. Matemáticamente se puede describir como:

$$Eliminar_{r_1 j_1 j_2} = P_{r_1} - S_{r_1 j_1 j_2} - c_{r_1 j_1} - c_{r_1 j_2} + K_{r_1 j_1 r_1 j_2}.$$

- $Sumar_{r_1 j_1 j_2 r_2 i}$: Este término representa el costo de añadir una subruta en una nueva posición dentro de la ruta r_2, otorgándole un nuevo valor al peso de esa ruta. Esta operación separa los clientes i e $i+1$ para insertar la subruta en esta posición, uniendo el inicio de la subruta con i y el final con $i+1$:

$$Sumar_{r_1 j_1 j_2 r_2 i} = P_{r_2} - c_{r_2 i} + S_{r_1 j_1 j_2} + K_{r_1 j_2 r_2 i} + K_{r_2 i r_1 j_1}.$$

Una vez definidas las restricciones y la función objetivo, se estructura el modelo de optimización.

1.3.5. El modelo de optimización

A continuación se presenta el modelo de optimización para la vecindad de reinserción de subruta.

$$\text{mín} \sum X_{r_1 j_1 j_2 r_2 i} \cdot (Eliminar_{r_1 j_1 j_2} + Sumar_{r_1 j_1 j_2 r_2 i} + penalty).$$

Sujeto a:

$$\sum X_{r_1 j_1 j_2 r_2 i} = 1.$$

En la siguiente sección se describe el modelo de optimización para la exploración de la vecindad de intercambio de subrutas.

1.4. Modelo de optimización para la vecindad de intercambio de subrutas

El intercambio de subrutas consiste en seleccionar dos subrutas de diferentes rutas y reubicar a cada una en la ruta de la otra. Para explorar esta vecindad, es necesario identificar las subrutas que se van a intercambiar. Esto implica seleccionar la ruta de la primera subruta, la ruta de la segunda subruta, identificar los clientes que conforman cada subruta y intercambiar la posición de las subrutas. Debido a que el número de rutas y la cantidad de clientes en las soluciones son conocidos, es posible formular este proceso como un problema de optimización.

El modelo de optimización que se presenta en esta sección utiliza los mismos parámetros definidos en las secciones 1.2 y 1.3. Sin embargo, las variables de decisión y la forma en la que se calculan el factor de penalidad *penalty* y el término *Sumar* difieren de los modelos anteriores debido a la naturaleza del intercambio de subrutas.

En el siguiente epígrafe se detallan las variables del modelo de optimización

1.4.1. Variables del modelo

Las variables para la exploración de esta vecindad se denotan como $Y_{r_1 j_1 j_2 r_2 i_1 i_2}$, donde j_1 y j_2 son los arcos que delimitan a una subruta de la ruta r_1, i_1 e i_2 son los arcos que delimitan a la subruta que pertenece a la ruta r_2, con la que se intercambia la primera subruta. Las variables $Y_{r_1 j_1 j_2 r_2 i_1 i_2}$ son variables binarias que toman los siguientes valores:

$$Y_{r_1 j_1 j_2 r_2 i_1 i_2} = \begin{cases} 1, & \text{si la subruta que comienza en el cliente } j_1 + 1 \text{ y termina en} \\ & \text{el cliente } j_2 \text{ de la ruta } r_1 \text{ se intercambia por la subruta de la ruta} \\ & r_2 \text{ que comienza y termina en los clientes } i_1 + 1 \text{ e } i_2 \text{ respectivamente,} \\ 0, & \text{en otro caso.} \end{cases}$$

En el caso del intercambio de subrutas, deben cumplirse restricciones adicionales para garantizar la validez de la operación y preservar la estructura del grafo de la solución. Cuando ambas subrutas pertenecen a la misma ruta, es decir $r_1 = r_2$, las subrutas no deben poseer aristas en común, debido a que durante el intercambio dichas aristas se duplicarían. Por lo tanto, las variables deben cumplir que $i_1, i_2 < j_1$ o $i_1, i_2 > j_2$ (Figura 1.9).

$0 \xrightarrow{i_1} 1 \to 4 \xrightarrow{i_2} 5 \xrightarrow{j_1} 2 \to 3 \xrightarrow{j_2} 0$ $\qquad$ $0 \to 2 \to 3 \to 5 \to 1 \to 4 \to 0$

Figura 1.9: Grafo de la solución antes y después del intercambio de la subruta [1,4] con la subruta [2,3].

Para ilustrar un ejemplo de cómo se modifica una solución utilizando las variables $Y_{r_1 j_1 j_2 r_2 i_1 i_2}$, se muestra un ejemplo cuando esta toma valor 1. Cuando $Y_{1,1,3,2,2,4} = 1$, indica un intercambio donde la subruta $[4, 5]$, formada por el segundo y tercer clientes de la ruta 1, se sustituye por la subruta $[8, 6]$ que posee al tercer y cuarto cliente de la ruta 2 (figura 1.10).

$r_1 : 0 \to 1 \to 4 \to 5 \to 7 \to 9 \to 10 \to 0$ $\qquad$ $r_1 : 0 \to 1 \to 8 \to 6 \to 7 \to 9 \to 10 \to 0$
$r_2 : 0 \to 2 \to 3 \to 8 \to 6 \to 0$ $\qquad$ $r_2 : 0 \to 2 \to 3 \to 4 \to 5 \to 0$

Figura 1.10: Reubicación de una subruta. Se resalta en color rojo la subruta que cambia de posición.

Como los parámetros ya se definieron en las secciones 1.2 y 1.3, a continuación se presenta el cálculo del factor de penalización *penalty* y la restricción de selección de una variable para la vecindad de intercambio de subrutas.

1.4.2. Restricciones

La restricción de selección de una variable se define como:

$$\sum Y_{r_1 j_1 j_2 r_2 i_1 i_2} = 1.$$

Para el intercambio de subrutas se calcula *penalty* de la siguiente manera.

$$penalty = \begin{cases} Y_{r_1 j_1 j_2 r_2 i_1 i_2} \cdot \alpha \cdot (RouteDemand_{r_2} - D_{r_2 i_1 i_2} + D_{r_1 j_1 j_2} - capacity) & (1) \\ Y_{r_1 j_1 j_2 r_2 i_1 i_2} \cdot \alpha \cdot (RouteDemand_{r_1} - D_{r_1 j_1 j_2} + D_{r_2 i_1 i_2} - capacity) & (2) \\ 0 & (3) \end{cases}$$

A continuación se explican los casos del cálculo de *penalty* para la exploración de la vecindad de intercambio de subrutas.

1. Si al eliminar la subruta delimitada por los arcos i_1, i_2 de la ruta r_2 e insertar la subruta delimitada por los arcos j_1 y j_2 en dicha ruta, la solución es infactible.

2. Si al eliminar la subruta delimitada por los arcos j_1, j_2 de la ruta r_1 e insertar la subruta delimitada por los arcos i_1 y i_2 en dicha ruta, la solución es infactible.

3. Si la solución es factible.

En el epígrafe siguiente se detalla el cálculo de la función objetivo incluyendo el factor de penalización.

1.4.3. Función Objetivo

El término $Eliminar_{r_1 j_1 j_2}$ se calcula de la misma forma en la que se calculó en la sección 1.3, debido a que para intercambiar dos subrutas se deben eliminar las dos y en la sección se definió como eliminar una subruta. El término $Sumar_{r_1 j_1 j_2 r_2 i_1 i_2}$ se puede calcular como la eliminación de la subruta delimitada por i_1, i_2 de la ruta r_2 y la unión de los extremos de la subruta j_1, j_2 con los clientes desconectados al eliminar i_1, i_2 de r_2:

$$Sumar_{r_1 j_1 j_2 r_2 i_1 i_2} = P_{r_2} - Eliminar_{r_2 i_1 i_2} - K_{r_2 i_1 r_2 i_2} + S_{r_1 j_1 j_2} + K_{r_2 i_1 r_1 (j_1+1)} + K_{r_1 j_2 r_2 (i_2+1)}.$$

Por tanto la función objetivo se puede calcular como:

$$\text{mín} \sum Y_{r_1 j_1 j_2 r_2 i_1 i_2} \cdot (Eliminar_{r_1 j_1 j_2} + Eliminar_{r_2 i_1 i_2} + Sumar_{r_1 j_1 j_2 r_2 i_1 i_2} + Sumar_{r_2 i_1 i_2 r_1 j_1 j_2} + penalty).$$

Una vez definidas las restricciones y la función objetivo, se estructura el modelo de optimización.

1.4.4. El modelo de optimización

El modelo de optimización para el intercambio de clientes se formula como:

$$\text{mín} \sum Y_{r_1 j_1 j_2 r_2 i_1 i_2} \cdot (Eliminar_{r_1 j_1 j_2} + Eliminar_{r_2 i_1 i_2} + Sumar_{r_1 j_1 j_2 r_2 i_1 i_2} + Sumar_{r_2 i_1 i_2 r_1 j_1 j_2} + penalty).$$

Sujeto a:

$$\sum Y_{r_1 j_1 j_2 r_2 i_1 i_2} = 1.$$

La resolución de los problemas de optimización que se describen en este capítulo permite explora la vecindad completa sin necesidad de evaluar explícitamente cada solución posible. A diferencia de una búsqueda exhaustiva, donde se evalúa cada vecino de forma individual, los modelos de optimización aprovechan estructuras matemáticas que reducen el tiempo de exploración [1]. Cada variable en el modelo representa una posible solución dentro de la vecindad, lo que permite se seleccione el mejor vecino sin necesidad de comprobar todas las soluciones posibles.

En este capítulo se presentó un nuevo enfoque para explorar vecindades en un VRP, utilizando modelos de optimización lineal entera, lo cual es un problema NP-Duro. Debido a la complejidad que posee resolver estos problemas, se relaja el problema para que las variables sean variables continuas con valores entre 0 y 1.

Al convertir las variables de los modelos a variables continuas, se pueden utilizar los algoritmos para problemas de optimización lineal continua. Gracias a la eficiencia que presentan estos algoritmos [1], se pueden explorar y encontrar el óptimo de las vecindades para el problema continuo.

Idealmente, se esperaba que el óptimo del problema continuo estuviera cercano al óptimo del problema entero. Sin embargo, al realizar los experimentos, en todos los casos que se analizan, la solución obtenida era óptima para el problema entero.

En el siguiente capítulo se presenta una vecindad más compleja debido a que el número de vecinos se encuentra en el orden de $O(n^4)$, por lo que es necesario encontrar una manera eficiente de generar los datos.

Bibliografía

[1] Sophie Huiberts. Upper and lower bounds on the smoothed complexity of the simplex method. 2024.