# ENFOQUE DE GENERACIÓN DE COLUMNAS PARA EL PROBLEMA DE LOCALIZACIÓN Y RUTEO CON PICKUP AND DELIVERY

Thomas Capelle, Universidad de Chile, thomas.capelle@inria.fr
Cristián E. Cortés, Universidad de Chile, ccortes@ing.uchile.cl
Michel Gendreau, Ecole Polythechnique de Montreal, michel.gendreau@cirrelt.ca
Pablo A. Rey, Universidad Diego Portales, pablo.rey@udp.cl
Louis-Martin Rousseau, Ecole Polythechnique de Montreal, louis-martin.rousseau@cirrelt.net

## RESUMEN

En este trabajo formulamos un modelo de programación entera para el Problema de Localización y Ruteo con Pickup and Delivery (PLRPDP). Para modelar este problema se propone un esquema de generación de columnas y para el subproblema, se implementa un algoritmo de label-setting, que resuelve el problema de camino más corto, con restricciones de Pickup and Delivery y ventanas de tiempo. Además se propone un conjunto de heurísticas para acelerar este proceso. Para validar el modelo, se hace una implementación del esquema de generación de columnas, el cual se prueba en diferentes instancias, algunas ya existentes en la literatura actual, como también otras desarrolladas en este trabajo. Destacando dentro de estas ´ultimas, las instancias clusterizadas y las de tipo corredor, las cuales por su geometría especial hacen que la localización de los depots sea de suma importancia. También se hace un análisis detallado de como los costos de apertura de los depots inciden en la solución óptima. Finalmente se presentan resultados del rendimiento de nuestra implementación para cada una de las instancias.

*Palabras claves: pickup and delivery, localización, generación de columnas*

## ABSTRACT

In this paper we formulate an integer programming model for the Location and Routing with Pickup and Delivery Problem. We propose a column generation scheme, in which the subproblem is implemented through a label-setting algorithm for the shortest path with pickup and delivery and time windows problem. We also propose a set of heuristics to speed up this process. To validate the model, we implement the column generation scheme and test it for different instances, some existing in the current literature, while others were built ad-hoc to represent some realistic configurations observed. In this sense, clustered instances as well as others where the points are located along a corridor, make the location of the depots of great importance due to their special geometry. It also gives a detailed analysis of how the costs of opening depots affect the optimal solution. Finally, we present performance results of our implementation for each of the implemented instances.

*Keywords: pickup and delivery, location, column generation*

## 1. INTRODUCCIÓN

En la actualidad, las empresas logísticas requieren tomar decisiones cada vez más complejas para el logro más eficiente de sus objetivos; en esta labor, el diseño y operación en forma óptima de cada elemento de la cadena de suministro son decisiones que se vuelven cada vez más relevantes para todos los involucrados, incluyendo fabricantes, despachadores, operadores intermedios, distribuidores y clientes. Respecto de las decisiones relativas con la ubicación de los lugares de almacenamiento y las políticas de diseño de esos espacios, el manejo de la flota de vehículos y la entrega final de los productos, ya sean concretos o virtuales, se requiere un manejo apropiado de todas las variables para ser eficiente y poder ahorrar significativamente los recursos involucrados.

Una de las decisiones estratégicas más importante en el diseño de un sistema logístico es la ubicación de los *depots*. Encontrar las potenciales ubicaciones no es siempre una tarea fácil, y por lo general constituye el objetivo de un estudio previo. Una vez que el conjunto de ubicaciones posibles está determinado, los planificadores deben elegir un subconjunto de tales lugares, que cumpla con las especificaciones del sistema y que a la vez sea óptimo para alguna función objetivo escogida. La naturaleza de esta decisión puede ser múltiple: minimizar costos de instalación, maximizar cobertura, optimizar alguna medida de bienestar o una combinación de las anteriores.

Si bien la ubicación es determinante, la eficiencia del sistema depende también de otros factores; entre ellos, el ruteo de los vehículos está entre los más importantes. Por lo tanto, los planificadores, además de determinar la ubicación óptima de los *depots*, deben determinar la cantidad de vehículos asignados para el manejo más adecuado del producto y el diseño de rutas más eficientes. La familia de problemas que estudian este tipo de decisiones es llamada Problemas de Ruteo de Vehículos, (Toth and Vigo, 2002).

Generalmente, si ambos problemas -localización y ruteo- aparecen juntos en un mismo escenario, son estudiados y resueltos por separado. Esta tendencia en general se justifica por la gran dificultad de resolver del problema combinado. Dado que ambos problemas pertenecen a la clase $\mathcal{NP}-hard$, en la mayoría de los casos su combinación resulta en a un problema más complejo. Sin embargo, estudiar el problema de ruteo de vehículos fijando la localización de los *depots* con anterioridad, generalmente conlleva a soluciones sub-óptimas.

A pesar de esta dificultad, los avances en tecnología, recursos computacionales y desarrollo de algoritmos en la actualidad, además de resultados exactos obtenidos en resolver localización y ruteo simultáneamente (Berger et al., 2007), proveen suficientes herramientas para plantear el problema conjunto que se estudiará en el contexto de este artículo: el *Problema de Localización y Ruteo con Pickup and Delivery*. Este problema es una variante del problema clásico de Localización y Ruteo, pero considera que la estructura de las rutas de los vehículos respeta un esquema de *Pickup and Delivery*. Así, el primer problema que se debe estudiar es el *Pickup and Delivery Problem* (PDP), en que una empresa de transporte de pasajeros o carga, consta con una flota fija de $m$ vehículos con capacidades fijas $Q$ cada uno, (Savelsbergh and Sol, 1995). En este problema, cada cliente $i$ tiene una ubicación de orígen $i^+$ y una ubicación de destino $i^-$. Analíticamente, el PDP resuelve la manera óptima de rutear los vehículos con el fin de satisfacer la demanda. Esto consiste en recoger la carga del cliente $i$ en $i^+$ y transportarla hasta su destino $i^-$.

Un ejemplo de lo anterior son los sistemas de *transfers* al aeropuerto , donde cada cliente es re-

cogido en su residencia, y llevado al aeropuerto o viceversa. Otro servicio del tipo PDP son los buses especiales para adultos mayores y discapacitados, los cuales recogen a sus pasajeros y los asisten en el viaje. A estos servicios en la literatura se les conoce como *Paratransit* o *Dial-a-Ride*, (Cordeau and Laporte, 2003).

En la literatura se encuentran algunas variantes a este problema, incluyendo restricciones de ventanas de tiempo en los orígenes o destinos. Por ejemplo, (Desrosiers et al., 1986) resuelve el problema para el caso de un solo vehículo, por medio de un algoritmo de programación dinámica. Por otro lado, (Dumas et al., 1991) resuelve el problema de múltiples vehículos proponiendo un algoritmo exacto por medio de un esquema de generación de columnas. Más recientemente, (Ropke and Cordeau, 2009) proponen un algoritmo de *Branch and Cut and Price* que considera dos subproblemas para el algoritmo de generación de columnas: si las rutas son elementales o no lo son. Cotas inferiores son agregadas dinámicamente, lo que mejora la ejecución del esquema general de generación de columnas.

En los problemas de ruteo de vehículos, la función objetivo a optimizar depende del problema específico que se quiere resolver. En general se utiliza una combinación del costo de los usuarios y el costo de operación de los vehículos. Por lo general, el costo de los usuarios viene dado por los tiempos de viaje y espera. Cuando el problema es modelado con ventanas de tiempo de servicio como restricción, el costo a los usuarios no es relevante en la función objetivo. Esto es así, ya que las ventanas de tiempo imponen en la solución óptima una calidad de servicio exigida por el usuario. Por lo tanto, la función objetivo suele estar asociada solamente al costo de ruteo.

Cuando la demanda es conocida con anterioridad, la localización inicial de los vehículos que se usarán para servir la demanda de manera óptima, son variables relevantes en un contexto de planificación, en especial si la demanda no es homogénea a lo largo de la semana. Agregar la ubicación de los *depots* como variables, generaliza el modelo tradicional de ruteo de vehículos, en el cual los *depots* son fijos. (Berger et al., 2007) plantea un modelo que integra tanto la localización de los *depots* como el ruteo de vehículos simultáneamente -Problema de Localización y Ruteo (PLR)-, agregando a la función objetivo un costo por abrir un depot además del costo de ruteo. Se propone un esquema de generación de columnas que es capaz de resolver instancias de hasta 10 *depots* candidatos y 100 clientes.

Con lo anterior, el objetivo central de este trabajo es formular un modelo que integre el *Pickup and Delivery Problem* y la localización óptima de los *depots*, el cual llamaremos *Problema de Localización y Ruteo con Pickup and Delivery* (PLRPD), además de proponer un esquema de solución eficiente.

La herramienta principal que se utilizará para resolver este problema es modelarlo como un esquema de generación de columnas. Esto divide nuestro problema en dos partes, un problema maestro (similar al propuesto por (Berger et al., 2007)), y un subproblema, el cual tiene como propósito alimentar al problema maestro con soluciones factibles. Para nuestro caso, el subproblema se ve modificado por las restricciones propias del PDP.

## 2. PROBLEMA DE LOCALIZACIÓN Y RUTEO CON *PICKUP AND DELIVERY*

El Problema de Localización y Ruteo con *Pickup and Delivery* combina en un solo modelo, el problema de localización de los *depots* y el problema de rutear los vehículos de manera óptima respetando las restricciones propias de un problema de *Pickup and Delivery*.

### 2.1. Formulación del PLR-PDP

El Problema de Localización y Ruteo con *Pickup and Delivery* es una extensión del Problema de Localización y Ruteo. El problema consiste específicamente, en elegir un conjunto de *depots* y construir las rutas asociadas, de tal manera de minimizar el costo de instalación además del costo de ruteo, respetando que los clientes deben ser servidos por una única ruta. Se debe notar que cuando nos referimos al cliente $i$, esto consiste en recogerlo en su *pickup* $i^+$ y depositarlo en su *delivery* $i^-$. Se considera entonces la formulación:

**Formulación PLR-PDP**

$$\text{mín} \quad \alpha \sum_{j \in J} f_j X_j + \sum_{j \in J} \sum_{k \in P_j} c_{jk} Y_{jk} \tag{1}$$

$$s.a: \quad \sum_{j \in J} \sum_{k \in P_j} a_{ijk} Y_{jk} = 1 \qquad \forall i \in I \tag{2}$$

$$X_j - \sum_{k \in P_j} a_{ijk} Y_{jk} \geq 0 \qquad \forall i \in I, \forall j \in J \tag{3}$$

$$X_j \in \{0,1\} \qquad \forall j \in J \tag{4}$$

$$Y_{jk} \in \{0,1\} \qquad \forall j \in J, \forall k \in P_j \tag{5}$$

Esta formulación corresponde a la de un problema de localización y ruteo presentada en (Berger et al., 2007). La diferencia en nuestro caso es que las rutas asociadas a los *depots* satisfacen las restricciones del PDPTW. Esto no se refleja directamente en la formulación del problema maestro del PLR-PDP, si no que en la construcción de las rutas propiamente tal. El conjunto $P_j$ consiste en las rutas factibles asociadas al *depot* $j$. Para cada ruta $k \in P_j$ se satisfacen las siguientes condiciones:

- La ruta $k$ comienza y termina en el *depot* $j$
- Si el cliente $i$ es servido por la ruta $k$, entonces la precedencia es respetada, esto es; $i^+$ (el *pickup*) aparece antes en la ruta que $i^-$ (el *delivery*)
- Si el cliente $i$ es servido por la ruta $k$, entonces sus ventanas de tiempo en $i^+$ e $i^-$ son respetadas
- La carga del vehículo ($q$) es siempre menor a su capacidad ($Q$)

Consideramos $P = \{i^+ \ : \ i \in I\}$ al conjunto de los *pickups* y $D = \{i^- \ : \ i \in I\}$ el conjunto de los *delivery*s. Para cada nodo $l \in P \cup D$, denotaremos su ventana de tiempo $[a_l, b_l]$ y $q_l$ la demanda por ser transportada en cada nodo (notar que $q_{i^+} = -q_{i^-}$).

La formulación del PLR-PDP de esta manera, contiene un número exponencial de variables $(Y_{jk})$. Por lo tanto, para instancias de tamaño práctico, enumerar las rutas no es algo posible. En vez de esto, usaremos un esquema de *branch and price* para resolver instancias de nuestro problema.

## 2.2. Branch and Price para el PLR-PDP

El algoritmo de *branch and price* que se propone está basado en el trabajo de (Berger et al., 2007), el cual es usado para resolver el PLR. Este método ha sido utilizado satisfactoriamente para resolver una gran variadad de problemas de *scheduling* y de ruteo de vehículos

Para generar un esquema de *branch and price* para el PLR-PDP usaremos el problema maestro PM definido por las ecuaciones (1)-(5). Llamaremos LPM a la relajación lineal de este problema maestro.

El conjunto rutas $P_j$ para cada *depot* no es conocido; por lo tanto, para el esquema de *branch and price* se resuelve una restricción de LPM con un conjunto $P'_j \subset P_j$ para cada *depot* $j \in J$. Cada uno de los conjuntos $P'_j$ es generado independientemente, asegurando la factibilidad del problema total en todo momento. Se denotará RPM a la restricción del LPM, donde sólo un subconjunto de las variables de rutas son consideradas (todas las de localización son siempre consideradas)

**RPM**

$$\text{mín} \quad \alpha \sum_{j \in J} f_j X_j + \sum_{j \in J} \sum_{k \in P'_j} c_{jk} Y_{jk} \tag{6}$$

$$s.a: \quad \sum_{j \in J} \sum_{k \in P'_j} a_{ijk} Y_{jk} = 1 \qquad \forall i \in I \tag{7}$$

$$X_j - \sum_{k \in P'_j} a_{ijk} Y_{jk} \geq 0 \qquad \forall i \in I, \forall j \in J \tag{8}$$

$$X_j \geq 0 \qquad \forall j \in J \tag{9}$$

$$Y_{jk} \geq 0 \qquad \forall j \in J, \forall k \in P'_j \tag{10}$$

La construcción de $P'_j$ es realizada en el subproblema o *pricing problem*.

## 2.3. Subproblema

Para el esquema de *branch and price* es necesario identificar los subproblemas asociados al problema maestro PLR-PDP del esquema. En este caso, para cada *depot* $j \in J$ se debe resolver un subproblema independiente, el cual consiste en identificar rutas del conjunto $P'_j$. Cada uno de estos subproblemas es un problema de *pickup and delivery* con restricciones de capacidad y ventanas de tiempo muy similar al expuesto en (Dumas et al., 1991).

El subproblema o *pricing problem* asociado a cada *depot* $j$ puede ser modelado como un problema de camino más corto sobre una red auxiliar, con restricciones de precedencia, ventanas de tiempo y capacidad. Este tipo de problemas se conoce en la literatura como *Elementary Shortest Path with*

*Pickup and Delivery and Time Windows Constraints* ESPPDTWC, ver (Ropke and Cordeau, 2009). Para su solución, se implementó un algoritmo de tipo *label-setting* especializado. Se denota por $z_j^*$ al costo reducido de la solución del *pricing problem* para el depot $j$. Si $z_j^* \geq 0 \quad \forall j \in J$, la relajación lineal del problema (LPM) ha sido resuelto. De lo contrario, para cada $j \in J$ tal que $z_j^* < 0$ se añade la correspondiente columna al RPM y se reoptimiza.

### 2.4. Esquema de *branching*

Una solución óptima del LPM puede contener variables con valores no enteros. Aplicar sobre esta solución un esquema *branch and bound* estándar no funciona, pues el conjunto de columnas no contiene todas las variables. Cuando las variables son fijadas durante la ramificación y el LPM es reoptimizado, las variables duales pueden tener nuevos valores. Como resultado, una columna que no tuvo costo reducido negativo, ahora sí puede tenerlo. Por lo tanto, nuevas columnas deben irse generando a medida que se recorre el árbol. Esta idea da lugar a los algoritmos conocidos como *branch and price* que corresponden a algoritmos *branch and bound* en donde un esquema de generación de columnas es usado para resolver los problemas lineales correspondientes a cada nodo del árbol.

Para que algoritmo *branch and price* sea efectivo en la práctica es importante utilizar una estrategia adecuada de ramificación que sea compatible con el *pricing problem*. Para esto, utilizamos una estrategia basada en la utilizada por (Berger et al., 2007). Estos autores usan una estrategia donde se mantiene la estructura de camino más corto en el *pricing problem*. La estrategia fundamental consiste en dos reglas de ramificación para los dos tipos de variables. Siempre que es posible, se ramifica en una variables de localización $X_j$. Para estas variables, la dicotomía de ramificación convencional es adecuada: fijar la variable $X_j = 1$ es fijar la utilización del depot $j$ imponiendo 1 en su valor, y resolver el *pricing problem* para el depot $j$; fijar la variable en $X_j = 0$ se logra imponiendo todas las $Y_{jk} = 0$ para todas las correspondientes $k \in P_j'$ en el RPM y después resolver el *pricing problem*.

Si todas las variables de localización toman valores enteros, entonces se realiza el *branching* en las variables de ruta. Para esto se utiliza una estrategia basada en la propuesta por (Dumas et al., 1991) seleccionando la variable correspondiente a una ruta con valor más cercano a 1.

## 3. RESULTADOS COMPUTACIONALES

Esta sección describe los experimentos computacionales realizados con el algoritmo de *branch and price* propuesto por este trabajo. Diversas instancias son usadas para obtener los resultados, algunas de las cuales son variantes de las instancias originales encontradas en la literatura para el PDPTW y otras instancias son generadas para probar partes específicas del modelo. El algoritmo fue codificado en *Python* versión 2.7 para MAC OSX y todos los experimentos fueron realizados en un computador *Intel i7* (2.2 GHz). Como *LP-solver* se usó CPLEX 12.0.1 en conjunto con la interfaz de *Python* para CPLEX. Todos los experimentos tenían una limitación en su ejecución de 1 hora.

Las instancias consideradas son modificaciones de las instancias usadas en (Ropke and Cordeau, 2009). Estas instancias originales consideran un único *depot* localizado en el centro de un cuadrado. Las posiciones de los clientes son elegidas uniformemente en un cuadrado de $[0,50] \times [0,50]$. La carga de cada cliente es elegida aleatoriamente en el intervalo $[5,Q]$, donde $Q$ es la capacidad del vehículo. Se considera un horizonte $T = 600$ y cada ventana de tiempo tiene ancho $W$. Las ventanas de tiempo son construidas aleatoriamente para cada cliente $i$. En (Ropke and Cordeau, 2009), el objetivo principal consiste en minimizar el número de vehículos usados, por lo tanto, se agrega un costo de $10^4$ a cada arco que sale del depot. En el problema que resuelve este trabajo, el objetivo es minimizar el costo de localización simultáneamente con el de ruteo, por lo tanto no se considera este valor en todas las instancias. Para poder usar estas instancias para el problema de localización y ruteo con *pickup and delivery* es necesario agregar nuevos *depots*, los cuales son agregados uniformemente, manteniendo el depot central original. Se consideran instancias hasta con 7 depots. Todas las instancias usadas son variaciones de las instancias tipo AA de (Ropke and Cordeau, 2009). Así las instancias con 10 y 15 clientes, son sólo los primeros 10 o 15 clientes de las respectivas A o B originales. Para obtener instancias con $W = 30$ se modificó para cada $i \in \{1,....,N\}$ la correspondiente ventana de tiempo superior $\hat{b}_i = b_i - 30$.

Considerando la cantidad de *depots* $|J|$ y la cantidad de clientes $N = |I|$, el cuadro 3 resume las características de nuestras instancias. Además las figuras 1-3 entregan una representación gráfica de las instancias. Las instancias con menos de 7 *depots* sólo consideran los primeros *depots*. Para las gráficas se usará la notación $i^+$ e $i^-$ para representar los *pickup* y *delivery* respectivamente. Los nodos rojos representan la ubicación de los *depots* y los blancos los clientes.

| $\|J\|$ | $Q$ | $W$ | $N$ |
|---|---|---|---|
| 1 | 15 | 30 | 10 |
| 1 | 15 | 60 | 10 |
| 3 | 15 | 30 | 10 |
| 3 | 15 | 60 | 10 |
| 3 | 15 | 30 | 15 |
| 3 | 15 | 60 | 15 |
| 3 | 15 | 30 | 30 |
| 3 | 15 | 60 | 30 |
| 7 | 15 | 30 | 30 |
| 7 | 15 | 60 | 30 |
| 3 | 20 | 30 | 30 |
| 3 | 20 | 60 | 30 |
| 7 | 20 | 30 | 30 |
| 7 | 20 | 60 | 30 |

Cuadro 1: Descripción de las Instancias

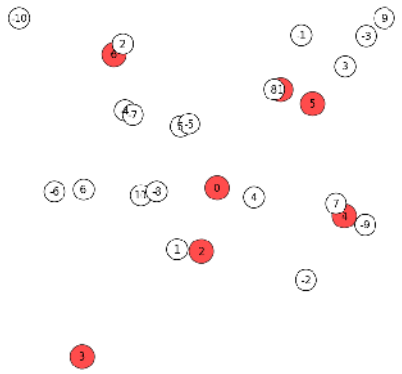

Figura 1: AA10: Instancia de 10 clientes

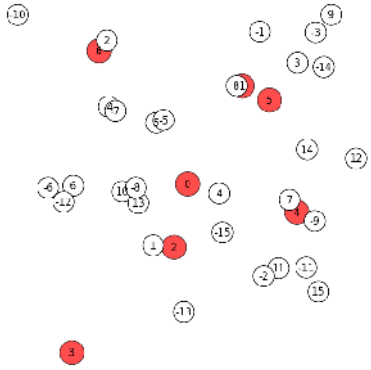

Figura 2: AA15: Instancia de 15 clientes

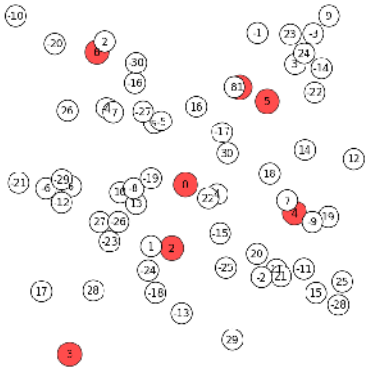

Figura 3: AA30: Instancia de 30 clientes

### 3.1. Análisis de Sensibilidad en el Costo de los Depots

La función objetivo del LPRPDP (1), tiene un parte asociada a los costos fijos de los *depots*. El valor de este costo fijo, incide directamente en el tipo de soluciones que se obtienen. Lo lógico, es que a mayor sea el costo fijo de abrir un *depot*, la solución óptima utilice menor cantidad de *depots*. La métodogía para realizar este análisis consiste en construir 30 instancias similares, y analizar el promedio de las soluciones. Para construir cada una de estas instancias, se consideran fijas las ubicaciones de los *depots* y las ventanas de tiempo de los clientes. La posición de los clientes son elegidas aleatoriamente respetando que el *pickup* y *delivery* se puedan satisfacer dentro de su ventana de tiempo. Para este análisis solo se consideró instancias de 30 clientes. La figura 4 muestra la utilización de *depots* en la solución óptima, a medida que se sube su costo de apertura.

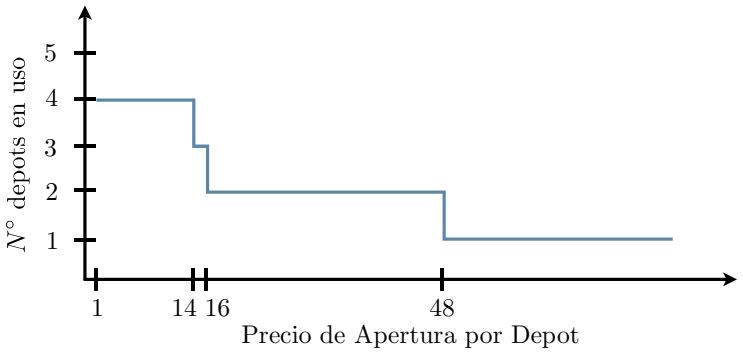


Figura 4: Uso de *depots* en solución óptima vs el costo unitario de apertura de cada *depot* para las instancias AA de 30 clientes

Para hacer más variado el análisis se considera también instancias similares a las de (Ropke and Cordeau, 2009) donde existe un costo de 10.000 unidades por utilizar cada vehículo. La idea de imponer este valor es minimizar la cantidad de vehículos usados en la solución óptima, implícitamente. (sin agregar esto en la función objetivo).

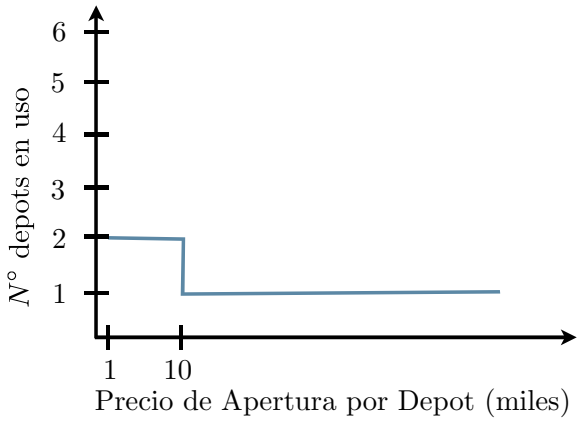


Figura 5: Uso de *depots* en solución óptima vs el precio costo fijo de apertura para las instancias AA de 30 clientes original de (Ropke and Cordeau, 2009)

En la figura 5 el análisis se torna interesante cuando el valor de abrir un depot supera el precio de una ruta (cada ruta tiene un costo base de 10.000 unidades). Hay que notar que en este caso se ven efectos cruzados, pues abrir un *depot* implica que al menos una ruta sale desde él, por lo tanto, la solución óptima trata de usar la menor cantidad de rutas posible, es decir, la menor cantidad de *depots*.

### 3.2. Instancias Especiales

La topología de las instancias es importante para la planificación tanto de la ubicación de los *depots* como de las rutas a utilizarse. En distintas situaciones tanto ciudadanas como extra urbanas, se encuentran organizaciones espaciales particulares. Por ejemplo en una ciudad como Santiago, donde cada comuna es como un gobierno regional, la necesidad de viajes intra comunales es alta. Por lo tanto, pensar en localizar centros de almacenamiento o transferencia en cada comuna es bastante razonable.

Desde un nivel macro, los viajes extra urbanos en Chile regional, pueden estudiarse como un corredor por el cual es necesario transportar los bienes a través de un eje central.

En esta parte se consideran 2 tipos de instancias. El primer tipo, son instancias *clusterizadas* o agrupadas (donde los clientes se encuentra agrupados por sectores), para ello, consideramos los clientes distribuidos en un cuadrado de $100 \times 100$, que a su vez dividimos en 4 cuadrantes de $50 \times 50$. Dentro de cada uno de estos cuadrantes repartimos los clientes. Una vez localizado el cliente $i$ en un cuadrante, tanto su *pickup* como *delivery* se localizan en él. Se consideran instancias con 30 clientes y 7 *depots*, donde los clientes son distribuidos aleatoriamente en los cuadrantes, y luego la posición de su *pickup* y *delivery* se distribuyen uniformemente en el cuadrante. Las ventanas

de tiempo usadas para estas instancias son construidas de la misma manera que en las instancias anteriores. Para aumentar la riqueza del análisis se construyen 30 instancias de esta manera. La figura 6 es un ejemplo de una de estas.

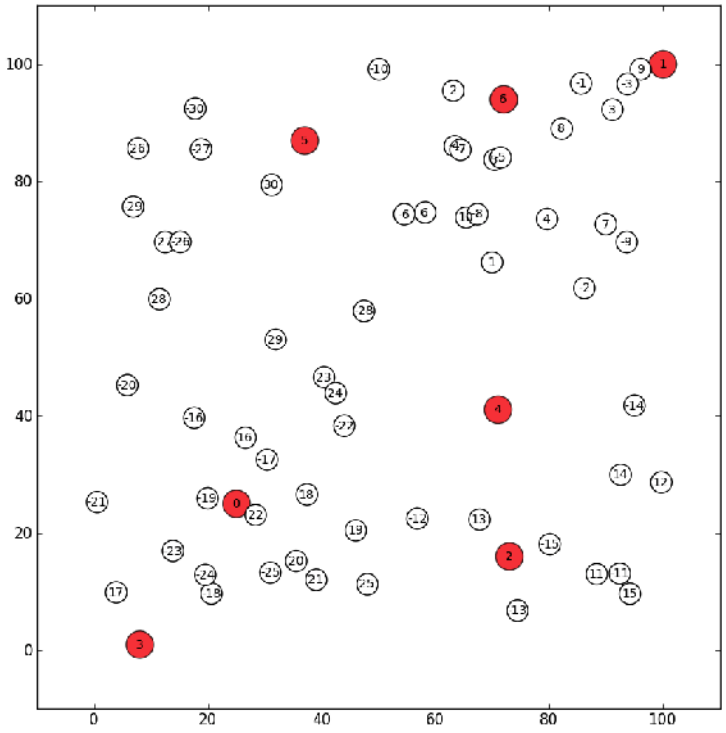


Figura 6: Instancia AA30C clusterizada

En este tipo de instancias, se repite el análisis de costos de los *depots*. Dado que los clientes se encuentran agrupados por sectores, esto fuerza de cierta manera la apertura de un *depot* por sector en la mayoría de los casos. la distribución de los clientes, el precio del *depot* incide menos en la solución obtenida. Ver figura 7

El otro tipo de instancias que se consideró, son instancias en las cuales los nodos se encuentran distribuidos a lo largo de un corredor. Aplicamos la misma metodología para construir estas instancias, solamente limitando a que los clientes sean repartidos en una faja diagonal de 30 unidades de ancho. La figura 8 representa una de estas instancias.

El análisis de costos (figura 9) entrega resultados esperables, siendo las instancias con geometría más difícil las más afectadas por el costo de los *depots*. Las instancias *clusterizadas* se ven muy afectadas por el precio del *depot*, puesto que no se puede servir a todos los clientes desde un *depot* central.

Las tablas 2 y 3 muestran los resultados relevantes de los modelos de *Branch and Price*, reportando: $Z_{IP}$ mejor solución entera, $Z_{LP}$ solución óptima de la relajación lineal, Gap: diferencia porcentual entre $Z_{IP}$ y $Z_{LP}$, Cpu Time LP: tiempo empleado en segundos en resolver los diferentes LPs, Cpu Time Pricing: tiempo empleado en segundos es resolver los subproblemas, Depots usados: cantidad de *depots* usados en la solución final, Nodos explorados: cantidad de nodos que exploró el esquema

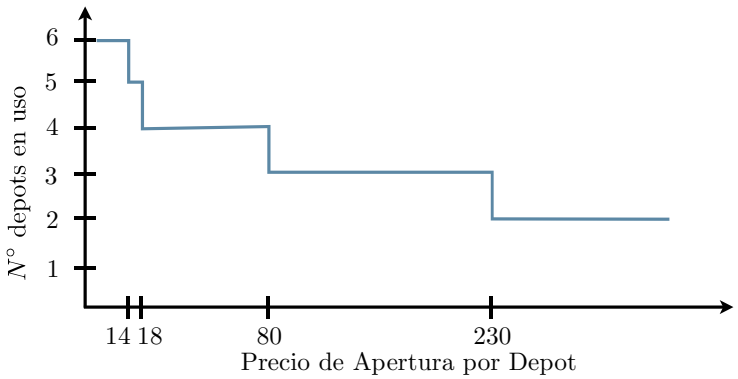


Figura 7: Uso de *depots* en solución óptima versus el precio costo fijo de apertura para las instancias clusterizadas AA30C

de *branching*, Columnas: cantidad de variables totales agregadas al problema maestro.

El primer cuadro 2 es el modelo aplicado a las instancias de 10, 15 y 30 clientes con tolerancia de 2 %, esto quiere decir que si el Gap es menor al 2 %, no se hace el *branching* y se entrega la solucíon en el nodo raíz con un *branch and bound* de CPLEX. La segunda tabla 3, es con Gap de 1 % sobre las instancias que originalmente no lo obtuvieron.

| Instancias | Q | W | $Z_{IP}$ | $Z_{LP}$ | Gap % | Cpu Time LP $[s]$ | Cpu Time Pricing $[s]$ | Depots usados | Nodos explorados | Columnas |
|---|---|---|---|---|---|---|---|---|---|---|
| AA10 | 15 | 30 | 343.37 | 343.37 | 0 | 0.10 | 8.99 | 3 | 1 | 415 |
| | 15 | 60 | 343.37 | 343.37 | 0 | 0.10 | 25.46 | 3 | 1 | 504 |
| | 20 | 30 | 343.37 | 343.37 | 0 | 0.09 | 10.50 | 3 | 1 | 415 |
| | 20 | 60 | 343.37 | 343.37 | 0 | 0.23 | 36.23 | 3 | 1 | 475 |
| AA15 | 15 | 30 | 483.92 | 477.99 | 1.23 | 0.21 | 45.65 | 5 | 1 | 958 |
| | 15 | 60 | 470.20 | 470.20 | 0 | 0.15 | 25.46 | 5 | 1 | 1114 |
| | 20 | 30 | 483.92 | 477.99 | 1.23 | 0.16 | 48.48 | 5 | 1 | 959 |
| | 20 | 60 | 470.20 | 470.20 | 0 | 0.33 | 71.39 | 5 | 1 | 1173 |
| AA30 | 15 | 30 | 1068.64 | 1068.64 | 0 | 1.21 | 132.55 | 5 | 1 | 6298 |
| | 15 | 60 | 1046.09 | 1043.31 | 0.26 | 3.27 | 241.84 | 5 | 1 | 17226 |
| | 20 | 30 | 1070.14 | 1059.15 | 1.03 | 1.10 | 151.49 | 5 | 1 | 6822 |
| | 20 | 60 | 1042.12 | 1031.67 | 1.01 | 4.03 | 291.52 | 6 | 1 | 17686 |

Cuadro 2: Resultados para todas las instancias, sin costo de *depot* con GAP¡ 2 %

| Instancias | Q | W | $Z_{IP}$ | $Z_{LP}$ | Gap % | Cpu Time LP $[s]$ | Cpu Time Pricing $[s]$ | Depots usados | Nodos explorados | Columnas |
|---|---|---|---|---|---|---|---|---|---|---|
| AA15 | 15 | 30 | 478.00 | 477.99 | 0.00 | 0.88 | 47.21 | 5 | 26 | 995 |
| | 20 | 30 | 478.00 | 477.99 | 0.00 | 0.70 | 51.55 | 5 | 26 | 995 |
| AA30 | 20 | 30 | 1068.64 | 1059.15 | 0.89 | 3.16 | 220.01 | 5 | 5 | 6845 |
| | 20 | 60 | 1033.27 | 1031.67 | 0.25 | 8.33 | 378.22 | 5 | 29 | 17717 |

Cuadro 3: Resultados para todas las instancias, sin costo de *depot* con GAP¡ 1 %

## 4. CONCLUSIONES

En este artículo se presenta una implementación del problema conjunto de localización de depots y pickup and delivery, que se plantea como un problema resuelto a resolver en forma simultánea , donde las decisiones de localización, así como las rutas de lickup and delivery son decisiones que

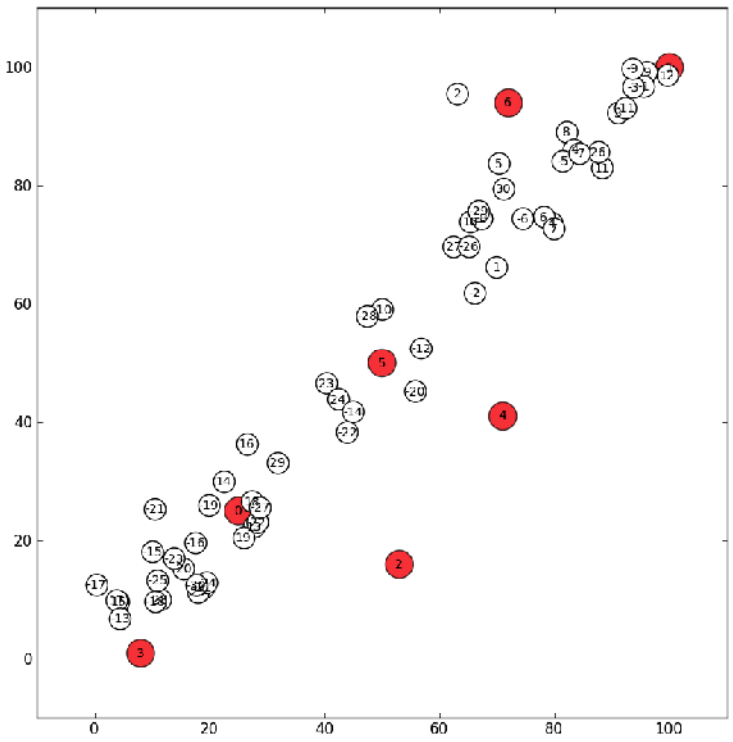


Figura 8: Instancia AA30Co tipo corredor

se consideran mutuamente para realizar mejores despachos y esquemas de ruteo desde una óptica de sistema. Se diseñó un esquema de branch and price para resolver este problema conjunto, el cual se implementó en instancias de distintos tamaños y configuraciones, algunas inspiradas en construcciones de la literatura, mientras que siguen configuraciones de sistemas realistas en base a clusterizaciones o configuraciones en esquemas de corredor.

La implementación del sistema conjunto es difícil, especialmente debido a la construcción del subproblema y la estrategia de branching considerada para moverse por el árbol del branch and bound. Se desarrolló un esquema de preproceso en base a heurísticas para acelerar la resolución en varias instancias relevantes, detalles de implementación que no se reportan en este artículo por restricciones de espacio. Los resultados tanto analíticos como numéricos son promisorios, mostrando una implementación directa de la combinación de dos problemas (localización de depots y pickup and delivery) que anteriormente fueron siempre tratados por separado; esquemas en conjunto de estos problemas sólo habían sido estudiados como decisiones independientes; en nuestro trabajo, se considera el problema formulado y resuelto de manera conjunta, lo cual dificulta su resolución, pero entrega soluciones donde realmente exista interacción entre las decisiones de localización de puntos de partida así como de configuración de rutas de tipo pickup and delivery, incluyendo por lo mismo todas las consideraciones necesarias que afecten a uno u otro problema.

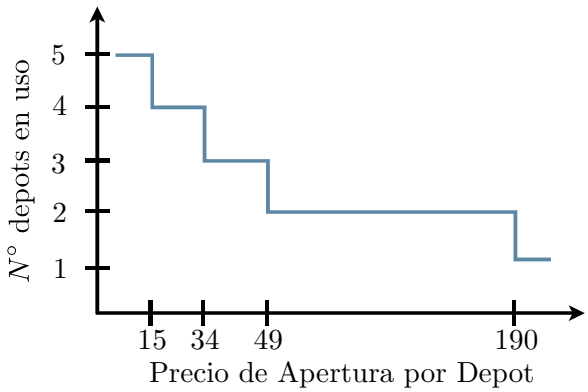


Figura 9: Uso de *depots* en solución óptima vs el precio costo fijo de apertura para las instancias de tipo corredor 30Co

## Agradecimientos

Los autores agradecen el apoyo financiero del Proyecto Fondecyt Chile 1100239, y del Instituto Milenio "Sistemas Complejos de Ingeniería" (ICM: P-05-004-F, CONICYT: 522 FBO16).

## Referencias

R Berger, C Coullar, and M Daskin. Location-routing problems with distance constraints. *Transportation Science*, 41(1):29–43, 2007.

Jean-Francois Cordeau and Gilbert Laporte. The dial-a-ride problem (darp): Variants, modeling issues and algorithms. *4OR*, pages 89–101, 2003.

Jaques Desrosiers, Yvan Dumas, and Francois Soumis. A dynamic programing solution of the large-scale single-vehicle dial-a-ride problem with time windows. *American Journal of Mathematics And Management Sciences*, 6(3):301–325, 1986.

Yvan Dumas, Jaques Desrosiers, and Francois Soumis. The pickup and delivery with time windows. *European Journal of Operational Research*, 54:7–22, 1991.

Stefan Ropke and Jean-Francois Cordeau. Branch and cut and price for the pickup and delivery problem with time windows. *Transportation Science*, 43(3):267–286, August 2009.

M.W.P Savelsbergh and M Sol. The general pickup and delivery problem. *Transportation Science*, 29(1):17–29, 1995.

Paulo Toth and Daniele Vigo, editors. *The Vehicle Routing Problem*. SIAM, 2002.