# OPTIMIZACIÓN REACTIVA PARA ITINERARIOS DE AVIONES CARGUEROS DE CORTO PLAZO

Julio Mora, Pontificia Universidad Católica de Chile jlmora@uc.cl
Felipe Delgado, Pontificia Universidad Católica de Chile fdb@ing.puc.cl

## RESUMEN

En este artículo se presenta un *Pickup and Delivery con ventanas de tiempo* (PDPTW) para optimizar las modificaciones en los itinerarios de aviones cargueros cuando ocurren cambios en la demanda de corto plazo. Para resolverlo se emplea la Descomposición de Dantzig Wolfe con Generación de Columnas, donde los subproblemas se resuelven con metaheuristica GRASP.. Los resultados obtenidos muestran que el algoritmo de solución propuesto muestras tiempos computacionales que son 400 veces más rápido que la solución exacta obtenida mediante solver tradicional a cambio de utilidades que son un 11% menores.

*Palabras claves: Pickup and Delivery, Itinerarios aéreos de carga, Generación de Columnas, metaheurística GRASP.*

## ABSTRACT

In this paper, a Pickup-and-delivery Problem with Time Windows (PDPTW) is presented to optimize the changes in the air cargo itineraries when the demand changes in the short term. The problem is resolved by mean of Dantzig Wolfe Decomposition with Column Generation, where each subproblems is solved using GRASP metaheuristic. The algorithm iterates till the best aircraft routing is found so that the utility of the whole fleet is maximized. Results show that the proposed solution algorithm achieve computational times that are 400 faster than the exact solution obtained by a traditional solver in exchange for profits that are 11% lower.

*Keywords: Pickup and Delivery, Air Cargo Schedule, Column Generation, GRASP metaheuristic.*

## 1. INTRODUCCIÓN

Una de las principales diferencias entre el transporte aéreo de carga y el de pasajeros es la alta incertidumbre de la demanda a ser transportada en cada itinerario. Esta incertidumbre se debe a que la demanda por carga se materializa pocas horas antes de ejecutarse los vuelos y generan que un importante número de aviones cargueros viajen con bajos factores de ocupación, lo que produce pérdidas inevitables para las aerolíneas.

Esta incertidumbre se explica porque las reservas son realizadas en ventanas de tiempo reducidas, lo que dificulta la estimación correcta de la oferta de aviones cargueros a proveer y se genera un desbalance entre la oferta planificada y las demandas reales. Más aún, ocurre frecuentemente que poco antes de efectuarse el vuelo, la carga reservada no se presenta o lo hace parcialmente (*No Show*), como también puede llegar fuera de plazo o a última hora. Por otro lado, no existen penalidades económicas para el cliente por no presentarse con la carga ni para la aerolínea por dejar carga *bookeada* abajo del vuelo.

Para afrontar este problema las aerolíneas realizan ajustes al itinerario de último minuto de manera manual, basándose en la experiencia de los tomadores de decisión. Estas modificaciones consideran medidas como: cancelación de vuelos, reruteo de aviones, volar aviones vacíos para reacomodar la carga (siempre cuando el análisis de rentabilidad lo permita), agregar escalas entre aeropuertos o efectuar vuelos de "ida-vuelta" (*roundtrip*); como también, una combinación de estas alternativas. Estas acciones son de tipo correctivas y locales de manera de impactar lo menos posible los turnos asignados a las tripulaciones afectadas.

El propósito de este trabajo es formular y resolver un modelo de programación entera mixta que permita optimizar los ajustes de itinerarios de corto plazo cuando ocurren disrupciones en la demanda, con el objetivo de maximizar las ganancias, minimizando los cambios realizados al itinerario base y sujeto a las restricciones operacionales propias de la industria del transporte aéreo.

## 2. REVISIÓN BIBLIOGRÁFICA

Nuestra revisión bibliográfica se centra en dos grandes temas. Primero, se analizan los trabajos y formulaciones relacionados a la planificación y recuperación de itinerarios aéreos tanto de pasajeros como de carga. Segundo, se abordan métodos de resolución para el *Pickup and Delivery con Ventanas de Tiempo* (PDPTW) que corresponde a la formulación propuesta.

### Planificación aérea

El estudio de la planificación aérea en el transporte de pasajeros ha sido ampliamente abordado (Abara, 1989; Rexing et al, 2000; Barnhart et al, 2003; Lohatepanont y Barnhart, 2004; Froyland y Maher, 2013). Sin embargo, con respecto al transporte de carga la literatura es más limitada. Derigs et al (2009) aborda el problema de la planificación estratégica de itinerarios cargueros mediante una formulación que simultáneamente optimiza la selección de vuelos, definidos previamente, la rotación de aviones e identifica las rutas de carga óptimas. Este enfoque da

énfasis a una mejora incremental de itinerarios mediante cambios en la selección de vuelos. Derigs y Frederichs (2012) hace una revisión general de la planificación aérea de carga y los subproblemas que esta conlleva para focalizarse en la etapa de diseño de itinerarios, donde a partir de una lista de vuelos prioritarios y opcionales, selecciona la combinación de vuelos más rentable. Yan y Young (1996) desarrollan un modelo estratégico que integra los procesos de asignación de flota con selección de vuelos para generar tablas horarias y rutas de aviones de pasajeros, cuando se espera que hayan cambios en las condiciones de la demanda de mercado en un futuro cercano. Yan et al. (2006) presentan un modelo que integra las etapas estratégicas de diseño de itinerarios y asignación de flota para aviones cargueros, que al ser resueltas en poco tiempo pueden ser empleadas en operaciones de corto plazo. Estos autores formulan el problema como un *Mixed Integer Multiple Commoditie Network Flow Problem* que busca integrar una red de vuelos a una red de carga y para esto se consideran todos los tramos de vuelo posibles, sin tener en cuenta una planificación base, costos adicionales por cancelar vuelos ni consideran el ruteo de los aviones.

A partir de lo anterior, se observa que el problema de ajustes a itinerarios cargueros de último minuto ante disrupciones en la demanda no ha sido abordado por la literatura a nivel operacional. Este problema tiene características especiales donde el tiempo es un recurso limitado, no se cuenta con una lista de vuelos definidas *a priori* y se requieren ajustes menores focalizados sobre los itinerarios de los aviones en la planificación base, de manera de afectar lo menos posible a la tripulación asignada a los itinerarios originales.

Métodos de solución para el *Pickup and Delivery con Ventanas de Tiempo* (PDPTW)

El problema presentado en este artículo se formulará como un *Pickup and Delivery con Ventanas de Tiempo* o PDPTW (Dumas et al., 1991; Cordeau et al., 2002; Parragh et al., 2008), con ajustes y restricciones adicionales para el transporte aéreo al momento de construir la red. El *Pickup and Delivery Problem* es una variación del *Vehicle Routing Problem* (Toth y Vigo, 2002), en que se busca generar una ruta para un vehículo que comienza y finaliza en una misma bodega central o en diferentes ubicaciones, y a lo largo de la ruta visita diferentes nodos recogiendo y entregando carga, en la medida que se respeten las restricciones de capacidad y ventanas de tiempo.

Este tipo de formulación en presencia de ventanas de tiempo es *NP-hard* (Savelsbergh y Sol, 1995), lo que implica que resolverlo mediante solución exacta sea altamente costoso en términos de tiempo. En el caso de nuestro problema el obtener una solución exacta pero en un periodo extenso de tiempo puede significar que sea imposible reaccionar de manera oportuna a los cambios que se producen a último momento en la demanda de carga.

Para solucionar este problema, existen diversos esfuerzos reportados en la literatura con excelentes resultados. En primer lugar, se han propuesto métodos exactos, como por ejemplo, Sigurd et al (2004) que emplea Generación de Columnas (Barnhart, 1998; Desrosiers y Lübbecke, 2005; Feillet, 2010; Venkateshan y Marthur, 2011) junto con descomposición de Dantzig-Wolfe (Dantzig y Wolfe, 1960) para el PDPTW. En su trabajo, para acelerar el tiempo de resolución de los subproblemas emplea un algoritmo de programación dinámica que permite resolver el problema a optimalidad con instancias de hasta 205 pedidos. En segundo lugar, métodos heurísticos de Búsqueda Local como por ejemplo *Simmulated Annealing* en Li y Lim (2001), *Tabu Search* en Nanry y Barnes (2000) y *Adaptive Large Neighborhood Search* en

Ropke and Pisinger (2006). En tercer lugar, métodos mixtos como Xu et al (2003) que propone un enfoque híbrido que combina la metodología de Generación de Columnas con heurísticas para resolver los subproblemas, lo que permite manipular instancias de hasta 500 pedidos. Este último enfoque se beneficia de la robustez de la solución exacta y la rapidez de la solución heurística para resolver problemas de gran tamaño. Debido a que nuestro problema es operacional y requiere de respuestas rápidas, se va a seguir un enfoque como el de Xu et al (2003) donde se logran obtener soluciones rápidas sin comprometer en demasía la calidad de estas.

El siguiente artículo está estructurado como sigue. En la siguiente sección, se plantea la formulación del problema como un modelo de programación entera mixta y luego se propone una formulación basada en la Descomposición de Dantzig Wolfe para ser resuelta mediante Generación de Columnas. En la Sección 4 se presenta el algoritmo de solución GRASP para resolver los subproblemas, junto con una metodología para acelerar su tiempo computacional, y se propone una heurística elaborada por los autores para transformar la solución óptima relajada en una solución óptima entera. En la Sección 5, se presentan los avances de los resultados computacionales para una instancia creada de cuatro aviones y un horizonte de planificación de tres días. Finalmente, las conclusiones y las perspectivas para futuros estudios se presenta en la Sección 6.

## 3. MODELACIÓN

El problema se modelará como un PDPTW, donde cada nodo corresponde a un evento con un rango de tiempo y aeropuerto definido, que simultáneamente puede representar un aterrizaje o despegue y carga o descarga. De este modo, los pedidos quedan definidos por un nodo *pickup* donde se carga el requerimiento de transporte y un nodo *delivery* donde se descarga, cada uno con su respectiva ventana de tiempo y duración de servicio.

Los arcos que conectan a estos nodos representan arcos terrestres si ambos nodos pertenecen a un mismo aeropuerto o tramos de vuelo si estos se encuentran en diferentes aeropuertos. Por otro lado, se definen diferentes subredes donde ciertos aviones pueden operar debido a restricciones operacionales o a derechos de vuelo que impiden que operen libremente en todos los aeropuertos de la red. Cada avión tiene asociado al inicio del horizonte de planificación su respectiva capacidad, aeropuerto de inicio y aeropuerto donde debe terminar al final del horizonte.

A continuación se presenta la notación del modelo para luego introducir la formación matemática.

### 3.1. Notación

**Conjuntos**

$K$: Conjunto de aviones identificados por su matrícula, $k = 1..|K|$.

$P$: Conjunto de pedidos, $P = \{1, \dots, n\}$. Sea $i \in P$, entonces el nodo *pickup* del pedido $i$ es $i^+$ y el nodo *delivery* $i^-$.

$P^k$: Conjunto de pedidos que pueden ser llevados por el avión $k$. $P^k \subseteq P$

$N^k$: Conjunto de nodos *pickup* y *delivery* que pueden ser visitados por el avión *k*, junto con los nodos de ubicación inicial y final del horizonte de planificación, $\{0_k{}^+, 0_k{}^-\}$. $N^k = P^k \cup \{0_k{}^+, 0_k{}^-\}$.

$NA_i^k$: Conjunto de nodos antecesores al nodo *i*, para el avión *k*, $NA_i^k = N^k \setminus \{i\} \wedge \{0_k{}^-\}$

$NS_i^k$: Conjunto de nodos sucesores al nodo *i*, para el avión *k*, $NS_i^k = N^k \setminus \{i\} \wedge \{0_k{}^+\}$

$A^k$: Conjunto de arcos que pueden ser recorridos por el avión k, $(i,j) \in A^k$, con $i, j \in N^k$

$S$: Conjunto de macro-nodos, que permiten consolidar nodos *pickup* y *delivery* consecutivos de un mismo aeropuerto, de modo tal que un macronodo define un nodo de origen o destino de un tramo de vuelo, independiente de los nodos *pickup* y/o *delivery* que se hayan visitado.

$S_n$: Conjunto de nodos ($\subseteq N$) que corresponden al desglose del macro-nodo $n \in S$.

$FB^k$: Conjunto de tramos de vuelo recorridos por el avión *k* en la planificación base, indexado como $(n,m)$ con $n, m \in S$.

**Parámetros**

$c_{i,j}^k$: Costo de recorrer el arco/vuelo $(i,j) \in A^k$ por el avión *k*. [US$/arco]

$t_{i,j}^k$: Tiempo de viaje entre el nodo $i$ y $j$ para el avión *k*. [hrs].

$d_i$: Tiempo de carga o descarga en el nodo $i \in N$. [hrs]

$e_i$: Instante de tiempo mínimo en que el nodo $i \in N$ puede ser visitado. [hrs]

$l_i$: Instante de tiempo máximo en que el nodo $i \in N$ puede ser visitado. [hrs]

$D_i$: Carga total demanda por el pedido $i \in P$. [toneladas]

$CAP^k$: Capacidad del avión $k$. [toneladas]

$s_{n,m}^k$: Costo de penalización por modificación del vuelo $(n,m)$ respecto a su planificación original para avión $k$. [$/vuelo]

$tarifa_i$: Tarifa por llevar el pedido $i \in P$. [US$/tonelada]

**Variables**

$x_{i,j}^k$: Variable binaria que toma valor 1 si el avión *k* recorre el arco $(i,j)$, 0 en otros caso.

$B_i^k$: Instante de tiempo en que el avión *k* inicia su visita al nodo $i \in N^k$. [hrs]

$q_i^k$: Cantidad de carga que puede ser recogida ($q_{i+}^k > 0$) o entregada ($q_{i-}^k < 0$) para el pedido $i \in P^k$ por el avión *k*. [toneladas]

$Q_i^k$: Carga que lleva el avión *k* al salir del nodo $i \in N^k$. [toneladas]

### 3.2. Modelo matemático

A continuación se presenta la modelación matemática del problema.

$$Max \sum_{k \in K} \sum_{i \in P^k} tarifa_{i^+}\, q_{i^+}^k - \sum_{k \in K} \sum_{(i,j) \in A^k} c_{i,j}^k x_{i,j}^k - \sum_{k \in K} \sum_{(n,m) \in FB^k} s_{n,m}^k \, max\{1 - \sum_{i \in S_n} \sum_{j \in S_m} x_{i,j}^k, 0\}$$

sujeto a:

$$\sum_{j \in P^k} x_{0_k{}^+, j^+}^k = 1 \qquad \forall\, k \epsilon\, K \tag{1}$$

$$\sum_{j \in NS_i^k} x_{i,j}^k - \sum_{j \in NA_i^k} x_{j,i}^k = 0 \quad \forall i \in N^k, k \epsilon\, K \tag{2}$$

$$\sum_{i \in P^k} x_{i^-, 0_k{}^-}^k = 1 \qquad \forall\, k \epsilon\, K \tag{3}$$

$$x_{i,j}^k = 1 \quad => \quad B_j^k \geq B_i^k + d_i + t_{i,j}^k \quad \forall (i,j) \epsilon\, A^k, k \epsilon\, K \tag{4}$$

$$x_{i,j}^k = 1 \quad => \quad Q_j^k \geq Q_i^k + q_j^k \qquad \forall (i,j) \epsilon\, A^k, k \epsilon\, K \tag{5}$$

$$Q_{0_k{}^-}^k = 0 \quad \forall\, k \epsilon\, K \tag{6}$$

$$\max\{0, q_i^k\} \leq Q_i^k \leq \min\{CAP^k, CAP^k + q_i^k\} \quad \forall i \in P^k, \forall\, k \epsilon\, K \tag{7}$$

$$q_{i^+}^k \leq CAP^k \sum_{j \epsilon\, NS_{i^+}^k} x_{i,j}^k \qquad \forall\, i^+ \in P^k, k \in K \tag{8}$$

$$\sum_{k \in K / i^+ \in P^k} q_{i^+}^k \leq D_i \qquad \forall i \in P \tag{9}$$

$$q_{i^+}^k + q_{i^-}^k = 0 \qquad \forall\, i \in P^k, k \in K \tag{10}$$

$$\sum_{j \epsilon\, NA_{i^+}^k} x_{j,i^+}^k - \sum_{j \epsilon\, NS_{i^-}^k} x_{i^-,j}^k = 0 \quad \forall i \epsilon\, P \subseteq N^k, k \epsilon\, K \tag{11}$$

$$B_{i^+}^k \leq B_{i^-}^k \quad \forall i \epsilon\, P \subseteq N^k, k \epsilon\, K \tag{12}$$

$$\mathrm{e_i} \leq B_i^k \leq \mathrm{l_i} \quad \forall i \epsilon\, N^k, k \epsilon\, K \tag{13}$$

$$x_{i,j}^k \epsilon\, \{0,1\} \qquad \forall (i,j) \epsilon\, A^k, k \epsilon\, K \tag{14}$$

$$q_i^k \in \mathbb{R} \quad \forall i \epsilon\, P^k, k \epsilon\, K \tag{15}$$

$$Q_i^k \geq 0 \quad \forall i \epsilon\, N^k, k \epsilon\, K \tag{16}$$

El objetivo del problema es de maximización de ganancias donde a los ingresos por llevar cada pedido se le restan los costos operacionales por operar con un determinado avión y los costos de penalización por modificación del vuelo de la planificación original, esta modificación puede ser una cancelación del vuelo o que el vuelo sea operado por otro avión. La restricción (1) restringe a que cada avión k comience en el nodo de inicio del horizonte de planificación. La restricción (2) asegura la continuidad entre nodos consecutivos de una ruta factible. La restricción (3) restringe a que cada avión *k* finalice su recorrido en el nodo final definido para el horizonte de planificación. La restricción (4) asegura que de ser recorrido un arco $(i,j)$ el instante de llegada al nodo j debe ser mayor o igual a la suma del instante de llegada en el nodo predecesor más el tiempo de carga o descarga en dicho nodo más el tiempo de viaje entre ambos nodos. La restricciones (5), (6) ,(7), y (8) aluden a las restricciones de capacidad asociadas al avión k. La restricción (9) limita la

cantidad máxima que puede ser recogida en un determinado nodo *pickup*. Con la restricción (10) se asegura que la carga recogida en un nodo *pickup* debe ser de igual tamaño que la entregada en su respectivo nodo *delivery*. En (11) se restringe que si un avión recorre un nodo *pickup* entonces debe visitar el nodo *delivery* de ese pedido. Con (12) se asegura que el nodo pickup sea visitado antes que el nodo *delivery*. La restricción (13) limita los instantes de arribo a cada nodo en base a sus ventana de tiempo. Finalmente, (14)-(16) corresponde a las restricciones para variables binarias y continuas.

El problema anterior es no lineal en el último término de la función objetivo y en el conjunto de restricciones (4), (5) y (7). Lo primero se puede corregir pasando como restricción este término y utilizando una variable binaria auxiliar $\mu_{n.m}$, que toma valor 1 si el tramo de vuelo $(n, m)$ se cancela y 0 si no. Lo segundo es fácilmente linealizable haciendo ciertas transformaciones a la formulación *big-M* propuesta por Cordeau (2006).

El problema planteado es *NP-Hard*, por tanto se recurre a Descomposición de Dantzig-Wolfe para reformular este problema como es expuesto en la próxima sección.

### 3.3. Descomposición de Dantzig Wolfe y Generación de Columnas

El problema planteado anteriormente tiene estructura de bloques, por lo tanto para la resolución se emplea la Descomposición de Dantzig-Wolfe (Dantzig y Wolfe, 1960), que consiste en dividir la formulación en un Problema Maestro Restringido (PMr) y un conjunto de Subproblemas (SPs), donde cada uno genera rutas factibles para un avión en particular. Luego, la relajación LP del Problema Maestro junto con los Subproblemas se resuelven en base a un enfoque de Generación de Columnas (Wolsey, 1998; Barnhart et al, 1998; Desrosiers y Lübbecke, 2005; Feillet, 2010; Venkateshan y Marthur, 2011) para determinar la ruta óptima que cada avión debe ejecutar, de manera de maximizar las ganancias con los menores cambios posibles a la planificación base y sujeto a las restricciones de demanda, capacidad y ventanas de tiempo.

El procedimiento de Generación de Columnas considera que en cada iteración se comienza con una relajación LP del PMr que al ser optimizado entrega valores duales que son utilizados en cada uno de los SPs para determinar los costos reducidos de cada ruta. Estos SPs se resuelven y si generan rutas que tienen costos reducidos mayores a 0 se ingresan al PMr. El algoritmo itera hasta que no sea posible ingresar nuevas columnas, es decir, todas las rutas generadas en la iteración tienen costo reducido no positivo.

A continuación se presenta la formulación y se profundiza el Problema Maestro Restringido y el *k*-Subproblema.

### 3.4. Problema Maestro Restringido (PMr)

#### Conjuntos

$R_k$: Conjunto de rutas factibles para el avión *k*.
$N^r$: Conjunto de nodos visitados por la ruta *r*.

**Parámetros**

$Y^k$: Representa la solución óptima del k-subproblema. Vector que se define como sigue:

$$Y^k = (\{x_{i,j}\}_{(i,j):i,j \in N^r}, \{q_i\}_{i \in N^r}, \{Q_i\}_{i \in N^r}, \{B_i\}_{i \in N^r}, \{\mu_{n.m}\}_{(m.m) \in FB^k})$$

$C^k$: Ganancia asociada a la ruta *r* del avión *k*. Vector que se define como sigue:

$$C^k = (\{c_{i,j}\}_{(i,j) \in r}, \{-tarifa_i\}_{i \in P^k}, \{0\}_{i \in D^k}, \{0\}_{i \in N^r}, \{0\}_{i \in N^r}, \{s_{n.m}\}_{(m.m) \in FB^k})$$

De este modo, $C^k Y^k$ se define como el valor del término de la función objetivo original para el avión *k* (Sección 2), evaluada en la solución o ruta generada por el *k*-Subproblema.

$q_{i,r}^k$: Carga del nodo *i* asignada al avión *k* en la ruta *r*, forma parte del vector $Y^k$.

**Variables**

$x_r^k$: Variable binaria que toma valor 1 si la ruta *r* se asigna al avión *k*.

**Modelo matemático**

$$Max \sum_{k \in K} \sum_{r \in R_k} (C^k Y^k)_r \, x_r^k$$

s.a.

$$\sum_{k \in K} \sum_{r \in R_k} x_r^k q_{i,r}^k \leq D_i \qquad \forall i \in P \quad (\pi_i) \tag{18}$$

$$\sum_{r \in R_k} x_r^k = 1 \qquad \forall k \in K \quad (\gamma_k) \tag{19}$$

$$x_r^k \in \{0,1\} \qquad \forall r \in R_k, \forall k \in K \tag{20}$$

El objetivo de este problema puede ser escrito como una combinación lineal convexa de los valores de las soluciones o rutas generadas por los subproblemas. De esta forma, las $R_k$ soluciones de los subproblemas pueden ser usados para generar una solución factible del problema original. La restricción (18) es vinculante a todos los subproblemas y corresponde a la restricción de demanda para cada nodo *pickup i,* relacionada con la restricción (8) del problema original. La restricción (19) es conocida como Restricción de Convexidad al corresponder a una combinación convexa de los puntos extremos o soluciones de cada subproblema. Ésta también se puede interpretar como una restricción que asegura que a cada avión *k* se le asigne solo una ruta *r* del conjunto de rutas factibles a operar por ese avión. La restricción (20) corresponde a la naturaleza binaria de la variable de decisión, sin embargo, debe ser relajada para facilitar la resolución del problema y así poder obtener las valores duales que se ingresan a los K-Subproblemas.

En cada iteración se obtienen los valores duales $\pi_i$ para cada nodo pickup de la red y $\gamma_k$ para cada *k*-subproblema. Cada $\pi_i$ se interpreta como el costo de oportunidad de llevar una unidad adicional del pedido *i* ante que el resto de los pedidos, este valor será igual para todos los *k*-subproblemas, a diferencia de los valores $\gamma_k$. Cada $\gamma_k$ representa la utilidad de la ruta óptima generada hasta ese momento para el avión *k*.

### 3.5. *k*-Subproblema o Problema Satélite

Cada problema satélite corresponderá a encontrar el ruteo óptimo para un avión en particular y está formado por el resto de las restricciones del problema original. La diferencia está en la función objetivo que ahora representa los costos reducidos del problema original, por lo tanto incluye los valores duales proporcionados por el Problema Maestro Restringido.

$$Max \left( \sum_{i \in P} tarifa_i \, q_{i^+} - \sum_{(i,j) \in A} c_{i,j} x_{i,j} - \sum_{(n,m) \in FB^k} s_{n,m} \, \mu_{n,m} \right) - \sum_{i \in P} \pi_i \, q_{i^+} - \gamma$$

sujeto a:

(1) - (8) y (10) -(16)

La solución obtenida por este problema es una ruta factible para el avión *k*, sin considerar las restricciones de demanda. Si el valor de la función objetivo o costo reducido de esta solución es mayor a 0, convendría incorporar esta solución al Problema Maestro Restringido.
El algoritmo de solución finaliza cuando todos los costos reducidos de los *k*-Subproblemas son no-positivos, es decir, ninguna columna se incorpora al Problema Maestro Restringido.

Debido a que cada subproblema supone resolver un *Pickup and Delivery* para cada avión se implementó una metaheurística GRASP (Feo y Resende, 1995) para resolver cada uno de estos y así acelerar el tiempo de resolución del mismo.

## 4. METODOLOGÍA DE RESOLUCIÓN

En este capítulo se presentará la metodología de solución. En primer lugar se describe la metaheurística GRASP (*Greedy Randomized Adaptive Search Procedures*) para luego explicar con más detalle la metodología de aceleración del algoritmo y finalmente la heurística de mezcla para transformar una solución óptima relajada a una entera.

### 4.1. Metaheurística GRASP.

El algoritmo GRASP tiene la siguiente estructura general (Talebian y Salari, 2015):

**Algoritmo 1.-** Esquema general de la estructura del algoritmo GRASP

```
MejorSolucion := ø;
MejorUtilidad = 0;
while (No se cumple criterio de término) do
    SolucionActual := Inicializacion();
    SolucionActual := LocalSearch(SolucionActual);
    if(Utilidad(SolucionActual) > MejorUtilidad) then
        MejorSolucion := SolucionActual;
        MejorUtilidad := Utilidad(SolucionActual);
    end
end
```

De esta manera, el algoritmo se divide en dos grandes etapas: Una primera etapa de Inicialización y luego una de Búsqueda Local o *Local Search.* Para la fase de Inicialización se construye una ruta inicial factible desde cero. Para la fase de Búsqueda Local, se emplean heurísticas, basadas en las propuestas por Cherkesly et al (2015) y Talebian y Salari (2015), para modificar y mejorar esta ruta inicial, a partir de la inserción de nuevos pedidos, intercambio de pedidos sin asignar con los asignados a la ruta y reubicación de los pedidos dentro de la ruta.

Cada modificación que se realiza tanto en la fase de Inicialización como en Búsqueda Local considera una función *greedy* que corresponde a la función objetivo del *k*-subproblema (Sección 2.5). Los valores duales de esta función se definen en cada iteración de la Generación de Columnas y se ingresan como parámetros al método que ejecuta el algoritmo GRASP.

### 4.2. Metodología de aceleración para GRASP

Para acelerar el tiempo de resolución del algoritmo se consideraron los siguientes mecanismos:

1. **Elección aleatoria de las heurísticas de Búsqueda Local:** En base a la eficiencia que tiene cada heurística para mejorar la función objetivo contra el tiempo de ejecución de la rutina se define una probabilidad de elección. En caso de que la heurística no mejore la función objetivo se procede a bloquear el acceso de esta en una futura iteración y se mantendrá bloqueada hasta que cualquiera otra heurística mejore la función objetivo.
2. **Considerar la planificación base como ruta del proceso de Inicialización:** Si bien mejora los tiempos al partir sobre una ruta base, se considera que este procedimiento se debe perfeccionar, debido a que en cada iteración del algoritmo GRASP se considera la misma planificación base.
3. **Paralelización de los Subproblemas:** Mediante programación se pretende paralelizar la ejecución de cada subproblema (algoritmo GRASP) a modo de reducir los tiempos computacionales sin perder la calidad de la solución.

### 4.3. Heurística de Mezcla

Como la solución obtenida al finalizar el algoritmo de Generación de Columnas con GRASP es relajada, nosotros elaboramos una heurística de mezcla. Esta heurística está basada en la misma estructura de GRASP, donde a partir de un conjunto de rutas óptimas relajadas de cada subproblema se obtiene una única ruta óptima para el respectivo avión.

El algoritmo de mezcla, consiste en tomar cada subproblema que tenga un conjunto de rutas óptimas y fijar la que tiene una utilidad ponderada mayor como ruta inicial. Luego en orden decreciente de utilidades ponderadas se van seleccionando el resto de las rutas. Para cada una de estas se revisa si los pedidos que visita fueron asignados a la ruta inicial. Si es así, su carga se suma a la carga ya asignada a ese nodo que tienen en común. Si no, se determina, si es posible, la mejor inserción para ese pedido, de forma tal que mejore la utilidad de la ruta. Finalmente, se obtiene solo una ruta óptima para cada uno de estos subproblemas.

## 5. RESULTADOS COMPUTACIONALES

El algoritmo fue programado en Python y se utilizó Gurobi 5.6.0 para resolver los modelos de optimización. El computador cuenta con cuatro procesadores Intel(R) Core (TM) i5-3210M, 2.5 GHz y con 8 GB de Memoria RAM.

A partir de la información proporcionada por una prestigiosa aerolínea de carga nacional se confeccionó una instancia ficticia consistente en un horizonte de planificación de 3 días. En esta red, se cuenta con 4 aviones B777, con capacidad de 100 toneladas cada uno. Las características de estos aviones se presentan en la Tabla 1.

**Tabla 1.- Características de la flota de aviones para instancia de prueba**

| Avión | Aeropuerto Origen | Aeropuerto Destino | Derechos de vuelo | Costo Variable [US$/Hr] |
|---|---|---|---|---|
| 1 | BSB | VCP | Todos | 10.000 |
| 2 | ASU | MIA | Todos | 10.000 |
| 3 | VCP | MIA | Todos | 10.000 |
| 4 | MIA | MIA | Todos | 10.000 |

Para esta instancia y debido a la falta de información, se consideró como supuesto que la red de pedidos coincide con la red de vuelos, es decir, el origen y destino de un vuelo será el origen y destino de esa carga transportada. La planificación base cuenta con 33 vuelos en total que visitan 15 aeropuertos y la demanda total a transportar es de 995 toneladas inicialmente. A esta planificación se le agregan 14 pedidos, que equivalen a un aumento del 30% en la demanda total del sistema, a modo de determinar una nueva planificación que maximice las ganancias y minimice los cambios.

Para probar la bondad del enfoque híbrido propuesto de Generación de Columnas con GRASP hemos comparado los resultados de este enfoque con los obtenidos mediante solución exacta (MIP). La solución exacta se resuelve mediante Gurobi, programa el cual realiza un Branch-and-Cut junto con algunas heurísticas para encontrar la solución óptima.

**Tabla 2.- Comparación resultados globales MIP y GRASP.**

| | MIP | GRASP |
|---|---|---|
| Valor F.O. (US$) | 594.902 | 535.205 |
| Best Bound (US$) | 651.388 | 651.388 |
| GAP (%) | 9.5% | 22% |
| Número de Variables | 23.344 | 16 |
| - Binarias | 22.388 | 16 |
| - Continuas | 956 | 0 |
| Número de Restricciones | 36.970 | 51 |
| Número de Non-Zeros | 205.830 | 188 |
| Tiempo Computacional (hrs) | 20,87 | 0,05 |

**Tabla 3.- Comparación resultados operacionales entre MIP y GRASP.**

| Avión | Cargas Transportadas (ton) | | | Factor de Ocupación | | Utilidades (US$) | | |
|---|---|---|---|---|---|---|---|---|
| | MIP | GRASP | Var. | MIP | GRASP | MIP | GRASP | Var. |
| 1 | 353,63 | 353 | 0,18% | 41% | 42% | 166.053 | 165.234 | 10,06% |
| 2 | 236,17 | 216,71 | 8,98% | 53% | 52% | 127.309 | 112.545 | 13,12% |
| 3 | 395,95 | 391,55 | 1,12% | 49% | 53% | 182.801 | 125.565 | 45,58% |
| 4 | 268,21 | 272,61 | -1,61% | 37% | 35% | 118.739 | 131.861 | 9,95% |
| **Total** | 1253,96 | 1233,87 | 1,63% | - | - | 594.902 | 535205 | 11,15% |

En la Tabla 2 se muestran los resultados globales para ambos algoritmos. En primer lugar se observa que mediante el MIP se obtiene un valor óptimo mayor al conseguido por GRASP. Sin embargo, el tiempo de resolución del MIP de 29,7 hrs es significativamente mayor a los 3 minutos que tarda GRASP. En segundo lugar, el gap de optimalidad presentado en la tabla es con respecto al *Best Bound* entregado por el MIP, el cual no necesariamente es el valor óptimo. Por lo tanto, si solo comparamos la utilidad final del MIP con el GRASP la diferencia es de un 11,15%. Aún cuando la utilidad obtenida por el GRASP en este caso es menor a la del MIP, los bajos tiempos computacionales del GRASP hacen que este enfoque de solución sea factible de ser utilizada en la práctica por los tomadores de decisiones a diferencia de lo que ocurre con el MIP.

Si se compara el número de variables y restricciones, se observa que GRASP con Generación de Columnas trabaja con un tamaño muy reducido del problema original. Esto demuestra que dicho algoritmo bien implementado podría ser más eficiente que el MIP y generar buenas soluciones en menor tiempo.

En términos de cargas transportadas, la Tabla 3 muestra que el enfoque GRASP transporta en total un 1.63% menos que el MIP. Esto se refleja en que el GRASP genere un 4.54% de demanda insatisfecha a diferencia del MIP que tiene solo un 2.98%. Con respecto a los factores de ocupación, en ambos algoritmos estos son cercanos al 50% para todos los aviones a excepción del avión 4 que presenta un bajo factor de ocupación de un 37% y 35% para el MIP y GRASP respectivamente. Esto se podría explicar por la baja demanda en cada pedido para los 3 días.

## 6. CONCLUSIONES

En este artículo, nosotros hemos planteado un nuevo enfoque para realizar los ajustes de último minuto a los itinerarios de aviones cargueros cuando ocurren modificaciones en la demanda de corto plazo. Este enfoque considera resolver un *Pickup and Delivery con Ventanas de Tiempos* mediante Generación de Columnas, donde los subproblemas son resueltos por una metaheurística GRASP.

Los resultados obtenidos hasta el momento permiten asegurar que el modelo planteado cumple con una finalidad operacional al generar itinerarios alternativos en un reducido tiempo computacional. El algoritmo de solución de Generación de Columnas y GRASP propuesto

muestra tiempos computacionales que son 400 veces más rápido que la solución exacta obtenida mediante *solver* tradicional a cambio de utilidades que son un 11% menores.

Como futuras líneas de investigación en el corto plazo se plantea mejorar el gap de optimalidad mediante la modificación en la fase de Inicialización y en la heurística para asignar cargas. A su vez, se plantea reducir aún más los tiempos de solución mediante la paralelización en la resolución de los subproblemas.

Además a futuro se plantea estudiar un enfoque de Branch-and-Price para resolver este problema y contraponer sus tiempos de resolución y calidad de la solución a los obtenidos mediante la heurística Mezcla propuesta en este artículo..

## 7. AGRADECIMIENTOS

Agradecemos al Fondo Nacional de Desarrollo Científico y Tecnológico por el financiamiento de este trabajo mediante el proyecto FONDECYT 11140436, como también a la aerolínea de carga por proveer información y opiniones valorables para el desarrollo de esta investigación.

## 8. REFERENCIAS BIBLIOGRÁFICAS

Abara J. (1989). Applying integer linear programming to the fleet assignment problem. **Interfaces** *19.*

Barnhart, C., E.L. Johnson, G.L. Nemhauser, M.W.P. Salvelsbergh, P.H. Vance. 1998. Branch-and-Price: Column generation for solving huge integer programs. **Operations Resarch**, 46, 316-329.

Barnhart, C., Belobaba, P., y Odoni, A. R. (2003). Applications of Operations Research in the Air Transport Industry. **Transportation Science**, 37(4), 307–324.

Cherkesly, M., Desaulniers, G., & Laporte, G. (2015). A population-based metaheuristic for the pickup and delivery problem with time windows and LIFO loading. **Computers and Operation Research**, *62*, 23–35.

Cordeau JF, Desaulniers G, Desrosiers J, SolomonMM, Soumis F (2002) VRP with time windows. En: Toth P, Vigo D (eds.) **The Vehicle Routing Problem**. SIAM, Philadelphia, PA, SIAM Monographs on Discrete Mathematics and Applications, vol. 9, 175–193.

Cordeau, J.-F. (2006). A Branch-and-Cut Algorithm for the Dial-a-Ride Problem. **Operations Research**, 54(3), 573–586.

Dantzig, G.B., y Wolfe, P. (1960). Decomposition Principle for Linear Programs. **Operations Research***,* 8(1), 101-111.

Derigs, U., Friederichs, S., y Schäfer, S. (2009). A New Approach for Air Cargo Network Planning. **Transportation Science**, 43(3), 370–380.

Derigs, U., y Friederichs, S. (2012). Air cargo scheduling: integrated models and solution procedures. **OR Spectrum**, 35(2), 325–362.

Desrosiers, J., y Lübbecke, M.E. (2005). **Column Generation**. 1st Ed. Springer, New York.

Dumas, Y., Desrosiers, J., y Soumis, F. (1991). The pickup and delivery problem with time windows, **European Journal of Operational Research**, 54, 7–22.

Feillet, D. (2010). A tutorial on column generation and branch-and-price for vehicle routing problems. **4or,** *8*(4), 407–424.

Feo, T., y Resende, M. (1995). Greedy Randomized Adaptive Search Procedures. **Journal of Global Optimization**, *6*(2), 109–133.

Froyland, G., y Maher, S. J. (2013). The Recoverable Robust Tail Assignment Problem, **Transportation Science**, 1–22.

Li H, Lim A (2001) A metaheuristic for the pickup and delivery problem with time windows. En: **13th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'01)**. IEEE Computer Society, Los Alamitos, CA, 333–340

Lohatepanont, M., y Barnhart, C. (2004). Airline Schedule Planning: Integrated Models and Algorithms for Schedule Design and Fleet Assignment. **Transportation Science**, *38*(1), 19–32.

Nanry WP, Barnes JW (2000). Solving the pickup and delivery problem with time windows using reactive tabu search. **Transportation Research Part B**, 34(2), 107–21.

Parragh, S. N., Doerner, K. F., y Hartl, R. F. (2008). A survey on pickup and delivery problems. **Journal Für Betriebswirtschaft**, 58(1), 21–51

Rexing , B. , Barnhart, C., Kniker, T., Jarrah, A., and Krishnamurthy, N. (2000) "Airline Fleet Assignment with Time Windows", **Transportation Science**, *34*, 1-20

Ropke S y Pisinger D (2006) An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. **Transport Science**, *40*, 455–472

Savelsbergh, M. W. P., y Sol, M. (1995). General pickup and delivery problem. **Transportation Science**, 29(1), 17.

Sigurd M, Pisinger D, Sig M (2004) Scheduling transportationof live animals to avoid the spreadof diseases. **Transport Science**, 38, 197–209

Talebian Sharif, M., y Salari, M. (2015). A GRASP algorithm for a humanitarian relief transportation problem. **Engineering Applications of Artificial Intelligence**, *41*, 259–269.

Toth, P., y Vigo, D. (2002). **The vehicle routing problem**. En: SIAM monographs on discrete mathematics and applications. Philadelphia: SIAM

Venkateshan, P., y Mathur, K. (2011). An efficient column-generation-based algorithm for solving a pickup-and-delivery problem. **Computers and Operations Research,** *38*(12), 1647–1655.

Yan, S., Chen, S.-C., y Chen, C.-H. (2006). Air cargo fleet routing and timetable setting with multiple on-time demands. **Transportation Research Part E: Logistics and Transportation Review**, 42(5), 409–430.

Yan, S., y Young, H. (1996). A Decision Support Framework for Multi-Fleet Routing and Multi-Stop Flight Scheduling. **Transportation Research Part A**, *30*(5), 379–398.

Wolsey, L. A. (1998). **Integer Programming**. 1st Ed. Wiley, New York.

Xu, H., Chen, Z.-L., Rajagopal, S., y Arunapuram, S. (2003). Solving a Practical Pickup and Delivery Problem. **Transportation Science**, *37*(3), 347–364.