Heurísticos GRASP híbridos para el problema de rutas de vehículos con restricciones de capacidad

Juan Guillermo Villegas R.

Tesis de Maestría en Matemáticas Aplicadas

UNIVERSIDAD EAFIT
ESCUELA DE CIENCIAS Y HUMANIDADES
DEPARTAMENTO DE CIENCIAS BÁSICAS
MEDELLÍN
2008

Heurísticos GRASP híbridos para el problema de rutas de vehículos con restricciones de capacidad

Juan Guillermo Villegas R.

Tesis de maestría presentada como requisito para optar al título de Magíster en Matemáticas Aplicadas

Asesor
Hermilson Velásquez C. PhD.

UNIVERSIDAD EAFIT
ESCUELA DE CIENCIAS Y HUMANIDADES
DEPARTAMENTO DE CIENCIAS BÁSICAS
Junio de 2008
MEDELLÍN

Agradecimientos

Al llegar al final de este proceso de formación quiero agradecer a las personas que contribuyeron para que se realizara con éxito:

Al profesor Hermilson Velásquez, asesor de mi tesis de maestría por el interés que siempre tuvo durante su desarrollo. Y sobre todo, por la confianza que siempre he recibido de él en muchos de mis proyectos académicos.

A Anna Sofía y María Clara, mi esposa e hija, los motores de mi vida, por acompañarme en cada proyecto, sin importar el lugar y el idioma.

A Andrés Mauricio, mi hermano y ahora colega, con quien siempre puedo discutir mis ideas y proyectos, y de quien siempre recibo valiosos comentarios y aportes.

A mi madre y a mi padre, por su apoyo incondicional…

Contenido

1. Introducción

El problema de rutas de vehículos con capacidad (VRP, *vehicle routing problem*) fue introducido en los 50's (Dantzig & Ramser, 1959); y desde entonces ha despertado la atención de los investigadores en matemáticas aplicadas e investigación de operaciones, ya que pertenece a los problemas NP-completos (Lenstra & Rinnooy Kan, 1981). De la misma manera también tiene importancia práctica debido a su aplicación en diferentes sectores de la economía; pues con el VRP o algunas de sus variantes se pueden modelar y optimizar la distribución en muchas industrias tales como la de bebidas, alimentos y otros productos de consumo masivo, entrega de periódicos y la recolección de residuos sólidos (Golden, *et al*., 2002). Además, las aplicaciones del VRP no se limitan a la recolección y distribución de bienes, también se utiliza para modelar la prestación de servicios como la reparación y entrega de electrodomésticos (Weigel & Cao, 1999), la atención médica domiciliaria (Miller & Weaver, 1997), las rutas de ventas (Jang, *et al*., 2006) y el transporte escolar (Corberán, *et al*., 2002). Para una revisión reciente del estado del arte en la investigación sobre VRP resultan útiles los trabajos de Laporte (2007) y Cordeau *et al*. (2007), así mismo, Hasle & Kloster (2007) presentan los elementos importantes en las aplicaciones prácticas del VRP.

Este trabajo tiene tres objetivos principales:

- El diseño e implementación de diferentes heurísticos híbridos basados en GRASP (*greedy randomized adaptive search procedures*) para la solución del VRP.
- El análisis de la influencia de los diferentes componentes de dicho método en su desempeño.
- La comparación de los resultados obtenidos con GRASP con respecto a los de otros métodos reportados en la literatura (Cordeau, *et al*., 2005).

En la segunda sección de este trabajo se presenta el VRP y los métodos más usados para su solución, a continuación, la tercera sección presenta las diferentes variantes de GRASP desarrolladas, la cuarta sección está dedicada a los experimentos computacionales realizados y su discusión, en la quinta y última sección se presentan las conclusiones y posibles extensiones de este trabajo.

2. Problemas de rutas con capacidad

El problema de rutas de vehículos con capacidad es el objeto principal de esta tesis. En su forma clásica el VRP busca un conjunto de rutas de mínimo costo; para atender un conjunto de clientes geográficamente dispersos, cuya demanda y localización se conocen, utilizando una flota (homogénea) compuesta por $\mathbf{K}$ vehículos con capacidad limitada, (en algunos casos también existe un límite sobre la distancia máxima que pueden recorrer). Cada cliente debe ser visitado por un solo vehículo, además, todas las rutas deben iniciar y terminar en un único nodo (el centro de distribución o bodega).

2.1. Formulación matemática

Sea $\mathbf{G(V, A)}$ un grafo, donde $\boldsymbol{V}$ es el conjunto de vértices (clientes) y $\boldsymbol{A}$ el conjunto de arcos. El vértice 0 está reservado para el centro de distribución o bodega, a cada arco $(\boldsymbol{i}, \boldsymbol{j})$ de $\boldsymbol{A}$ está asociado una cantidad $\mathbf{c_{ij} \geq 0}$, que representa el costo o distancia necesaria para ir del cliente $\boldsymbol{i}$ al cliente $\boldsymbol{j}$. Además, cada cliente tiene asociada una cantidad no negativa $\mathbf{d_i \geq 0}$, correspondiente a su demanda. Los vehículos tienen capacidad $\boldsymbol{Q}$, que limita la cantidad máxima de clientes que pueden atender.

Una de las formulaciones del VRP considera variables binarias:

$$x_{ijk} = \begin{cases} 1: si\ el\ arco\ (i,j) es\ atravesado\ por\ el\ vehiculo\ k = \{1,..,K\} \\ 0: en\ caso\ contrario \end{cases}$$

$$y_{ik} = \begin{cases} 1: si\ el\ cliente\ i\ es\ visitado\ por\ el\ vehiculo\ k = \{1,..,K\} \\ 0: en\ caso\ contrario \end{cases}$$

Con estas variables el problema se formula de la siguiente manera:

$$Min \sum_{i \in V} \sum_{j \in V} c_{ij} \sum_{k=1}^{K} x_{ijk} \quad (1)$$

sujeto a:

$$\sum_{k=1}^{K} y_{ik} = 1 \quad \forall i \in V - \{0\} \quad (2)$$

$$\sum_{k=1}^{K} y_{0k} = K \quad (3)$$

$$\sum_{j \in V} x_{ijk} = \sum_{l \in V} x_{lik} = y_{ik} \quad \forall i \in V, k = 1 \ldots K \quad (4)$$

$$\sum_{i \in V} d_i y_{ik} \leq Q \quad \forall k = 1 \ldots K \quad (5)$$

$$\sum_{i \in S} \sum_{j \in S} x_{ijk} \leq |S| - 1 \quad \forall S \subseteq V \setminus \{0\}, |S| \geq 2, k = 1 \ldots K \quad (6)$$

$$x_{ijk} \in \{0,1\} \quad \forall i, j \in V, k = 1 \ldots K \quad (7)$$

$$y_{ik} \in \{0,1\} \quad \forall i \in V, k = 1 \ldots K \quad (8)$$

La función objetivo (1) expresa el costo total de los arcos recorridos en la solución. Las restricciones (2) establecen que un cliente es atendido por un solo vehículo. La restricción (3) limita el número de vehículos que parten del centro de distribución haciendo que se utilicen K vehículos.

A continuación, las restricciones (4) establecen la conservación del flujo en todos los clientes haciendo que el vehículo asignado a un cliente entre y salga de este. Las restricciones (5) hacen que la demanda de los clientes asignados a un vehículo no exceda su capacidad. Por último aparecen las restricciones (6) de eliminación de subtour. Las variables binarias se definen en las restricciones (7) y (8).

Cuando se busca un método de solución para el VRP se tienen dos alternativas, la primera es utilizar métodos de enumeración implícita basados en *branch and bound.* Pero dado que el VRP es NP-completo (Lenstra & Rinnooy Kan, 1981), el tiempo computacional necesario para encontrar soluciones óptimas a problemas de tamaño moderado puede llegar a ser muy grande. En general, estos métodos solamente son efectivos para resolver problemas con menos de 200 clientes (Baldacci, *et al*. 2007). Esto justifica la utilización de métodos de solución alternativos, más aún cuando existen aplicaciones que pueden tener varios cientos de clientes cuando se aplica en la distribución de bebidas (Golden, *et al*., 1998), e incluso varios miles cuando se trata de recolección de residuos sólidos (Kytöjoki, *et al*., 2007).

Entonces, la segunda alternativa es la solución del VRP son los métodos aproximados de optimización, heurísticos o meta-heurísticos, que generan soluciones de muy buena calidad en un tiempo razonable. Las razones para utilizar métodos aproximados para el VRP son la rapidez con la cual se pueden obtener buenas soluciones, además de su sencillez y flexibilidad, muy importantes cuando se busca abordar situaciones prácticas (Cordeau, *et al*., 2002). Es importante resaltar que dichos métodos (heurísticos y metaheurísticos) son la estrategia de solución predominante en el software comercial de diseño de rutas (Hall & Partyka, 2008).

Entre los métodos heurísticos más importantes se encuentra el famoso algoritmo de los ahorros (Clarke & Wright, 1964), el algoritmo de barrido (Gillett & Miller, 1974) y el método de los pétalos (Foster & Ryan, 1976). Laporte & Semet (2002) reseñan ampliamente los métodos clásicos para la solución del VRP.

Por su parte los metaheurísticos evolutivos, las algoritmos de aceptación por umbral y la búsqueda tabú son los algoritmos metaheurísticos más exitosos en la solución del VRP, tal como se resume en la Tabla 1, en la cual la

columna *Desviación promedio MSC* reporta la desviación con respecto a la mejor solución conocida en experimentos con instancias públicas de prueba.

Tabla 1 Metaheurísticos para el VRP y su desempeño. Fuente: (Cordeau, *et al.*, 2005). Tomado de (Prins, 2006)

Tipo de Método	***Autores***	***Año***	***Desviación promedio MSC***
Búsqueda local guiada + Estrategia Evolutiva	Mester & Braysy	2004	0.03%
Búsqueda local guiada + Estrategia Evolutiva	Mester & Braysy	2004	0.07%
Memoria Adaptativa	Tarantilis & Kiranoudis	2002	0.23%
Algoritmo memético	Prins	2004	0.24%
Algoritmo de umbral	Li *et al*	2004	0.41%
Algoritmo memético	Berger & Barkaoui	2004	0.49%
Búsqueda tabú	Cordeau *et al*	2001	0.56%
Búsqueda tabú granular	Toth & Vigo	2003	0.64%

3. Métodos GRASP híbridos para el VRP

GRASP (abreviatura de *greedy randomized adaptive search procedures)* es un método metaheurístico que fue creado a finales de los 80 (Feo & Resende, 1989) para resolver problemas de optimización combinatoria. GRASP puede verse como una estrategia multiarranque (sin memoria) que construye soluciones de alta calidad que son mejoradas posteriormente usando búsqueda local (Martí, 2003). Una de las principales ventajas de GRASP es su sencillez y facilidad de implementación, ya que por lo general se tienen métodos constructivos y de mejora para los problemas de optimización que se quieran resolver.

GRASP ha sido aplicado exitosamente para resolver muchos problemas en diversas disciplinas (Festa & Resende, 2004), sin embargo, no ha recibido la suficiente atención como alternativa para resolver el VRP (Gendreau, *et al*., 2008), aunque en algunas variantes como el VRP con ventanas de tiempo (Kontoravdis & Bard, 1995) ha obtenido buenos resultados. De GRASP aplicado al VRP apenas se conocen unos pocos trabajos, que además tienen escasa difusión en la literatura (Snyder, 2003), (Hjorring, 1995), (Baker & Carreto, 2003).

Otra motivación importante para este trabajo es la posibilidad que ofrece GRASP para la creación de metaheurísticos híbridos con diferentes variantes, tales como: mejoras a la construcción aleatorizada, utilización de metaheurísticos como búsqueda tabú, recocido simulado o búsqueda de vecindario variable en la etapa de mejoramiento o la utilización de GRASP para generar soluciones iniciales que luego son operadas con reencadenamiento de trayectorias u otros métodos poblacionales (Resende, 2008). La implementación de algunos de estos híbridos de GRASP para la solución del VRP se describe en las próximas secciones.

El Algoritmo 1 presenta el procedimiento general de GRASP. En él se puede ver que los componentes principales de GRASP son la construcción y la mejora de las soluciones. `Max_Iteraciones` corresponde al número de iteraciones que se repetirá el ciclo de *construcción-mejora.* Y `parámetros` corresponde a los valores que controlan las diferentes variantes de GRASP (por ejemplo, la cardinalidad de la lista de candidatos, tal como se verá más adelante).

```
Procedimiento GRASP(Max_iteraciones, parámetros)
   1. Lectura de Datos
   2. Para k=1 hasta Max_Iteraciones Haga
   3.       Solucion←Construccion_Aleatorizada(Parámetros)
   4.       Solucion←Busqueda_Local(Solucion)
   5.       ActualizarMejorSolucion(Solucion,Mejor_Solucion)
   6. Fin Para
   7. Muestre  Mejor_Solucion
Fin GRASP
```

Algoritmo 1. Procedimiento general de GRASP. Adaptado de Resende & Ribeiro (2003)

3.1. Construcción Aleatorizada

En general, los métodos constructivos, encuentran una solución incorporando un elemento a la vez en cada paso del proceso de construcción, obteniendo así una solución parcial. Un elemento que pueda seleccionarse como parte de una solución parcial se llama elemento candidato.

Para determinar cuál elemento candidato incluir en la solución se hace uso de una función miope, que mide la contribución local de cada elemento a la solución parcial. Existen varias formas posibles de introducir aleatoriedad a este procedimiento.

Una de las primeras ideas fue el uso de una lista restringida de candidatos (RCL); tal lista contiene un conjunto de elementos candidatos con los mejores valores de la función miope, el siguiente candidato a ser agregado a la solución se selecciona al azar de la lista restringida de candidatos. Dicha lista puede consistir de un número fijo de elementos (restricción por

cardinalidad) o elementos con los valores de la función miope dentro de un rango dado. El pseudocódigo de este componente de GRASP se presenta en el Algoritmo 2.

```
Procedimiento Construccion_Aleatorizada()
   1. Solucion ← φ
   2. Evalué el costo incremental de los elementos candidatos
   3. Mientras Solucion no sea una solución completa Haga
   4.      Construya la lista restringida de candidatos (RCL)
   5.      Seleccione un elemento s de RCL aleatorimente
   6.      Solucion ← Solucion ⋃ {s}
   7. Fin Mientras
   8.  Muestre Solucion
Fin Construccion_Aleatorizada
```

Algoritmo 2. Construcción Aleatorizada GRASP. Adaptado de Resende & Ribeiro (2003)

A este procedimiento pueden introducírsele ciertas variantes, como sesgar la probabilidad de selección de los candidatos de la RCL o perturbar los datos. Estos procedimientos son descritos en (Resende & Gonzalez-Velarde, 2003).

El método de construcción utilizado en este trabajo es el método de los ahorros (Clarke & Wright, 1964), uno de los métodos constructivos más utilizados para resolver el VRP; debido a su sencillez, facilidad de implementación y velocidad. Para una adaptación de dicho método a situaciones reales, en software comercial y académico, véanse por ejemplo Poot, *et al*. (2002), Snyder (2003) y Faulin, *et al*. (2005).

El método de los ahorros puede explicarse de la siguiente manera: para cada par de clientes, (i, j) se calcula el ahorro, s_{ij}, que se generaría al incluirlos en una misma ruta, dicho ahorro es calculado con la expresión:

$$s_{ij} = c_{0j} + c_{0i} - \lambda * c_{ij} \qquad (9)$$

El valor de λ permite penalizar la distancia que existe entre clientes muy lejanos. La Figura 1 ilustra el cálculo de los ahorros.

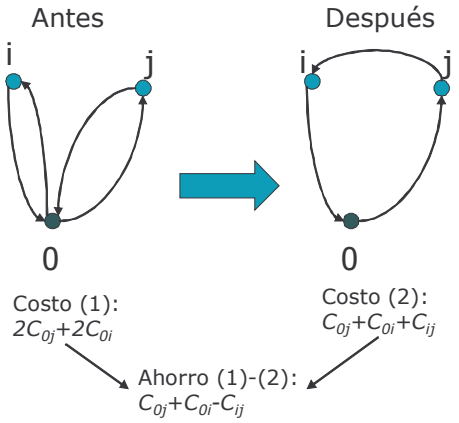


Figura 1 Método de los ahorros

Dichos ahorros son organizados en orden no decreciente y se procede a construir las rutas usando uno de las siguientes alternativas:

- la versión secuencial que considera una ruta a la vez
- la versión paralela que construye simultáneamente varias rutas uniendo rutas más pequeñas.

En este trabajo, se utiliza la versión secuencial del método de los ahorros como método de construcción aleatorizada, bajo un esquema de perturbación aleatoria, de esta manera es posible utilizar la versión secuencial del método de los ahorros sin modificación alguna, simplemente es necesario aleatorizar de cierta manera los valores de ahorros antes de aplicar el método constructivo; un procedimiento similar de aleatorización fue utilizado recientemente por Girard, et al. (2005).

La expresión de aleatorización de los ahorros es la siguiente:

$$s'_{ij} = r \times s_{ij} \qquad (10)$$

Donde:

α: parámetro que define la aleatorización, con valores en el intervalo [0,1]

s_{ij}: valor original del ahorro calculado usando la expresión (9)

s'_{ij}: valor del ahorro aleatorizado

r: número aleatorio uniforme en el intervalo $[1-\alpha, 1+\alpha]$.

3.2. GRASP Reactivo

Una de las características deseables de los método heurísticos es que tenga pocos parámetros y que los que existan tengan sentido para el usuario (Golden, *et al*., 1998). Según este enfoque, para simplificar el proceso relacionado con la afinación de los parámetro α y λ, se decidió utilizar una estrategia reactiva en la cual los valores de α y λ se adaptan dinámicamente según los resultados obtenidos en las iteraciones previas.

El procedimiento reactivo para la selección de α, introducido por Prais & Ribeiro (2000), se utiliza en este trabajo para controlar la magnitud de la perturbación aleatoria α y para seleccionar reactivamente λ.

En GRASP reactivo, en vez de utilizar valores fijos para α y λ se utiliza un conjunto discreto de valores predeterminados $A = \{\alpha_1, \alpha_2, ..., \alpha_m\}$; $L = \{\lambda_1, \lambda_2, ..., \lambda_m\}$ para α y λ, en cada iteración se selecciona aleatoriamente un valor para α y λ, que luego se utiliza en la ecuación de perturbación de los ahorros (10).

Sea p_i la probabilidad de seleccionar el valor α_i, para $i = 1,...,m_\alpha$ y p_j la probabilidad de seleccionar el valor λ_j, para $j = 1,...,m_\lambda$. Inicialmente, la función de masa de probabilidad es uniforme (en adelante solo se mostrarán las ecuaciones para α que se aplican de la misma manera para λ:

$$p_i = \frac{1}{m_\alpha},\ i = 1,...,m_\alpha \qquad (11)$$

Periódicamente, se actualizan las probabilidades p_i, utilizando la información del desempeño obtenido en las iteraciones anteriores al usar los diferentes valores de α_i.

Sean:

$C_t^*(s)$: distancia de la mejor solución conocida en la iteración t.

$\overline{C_i}$: distancia promedio de las soluciones obtenidas con el valor α_i.

El procedimiento utilizado para actualizar las probabilidades es el siguiente:

1. Para $i = 1,\ldots, m$, calcular:

$$q_i = \left(\frac{C_t^*(s)}{\overline{C_i}} \right)^{\delta}, i = 1,\ldots, m \qquad (12)$$

2. Normalizar los valores de q_i, para obtener los nuevos valores de p_i:

$$p_i = \frac{q_i}{\sum_{j=1}^{m} q_j} \qquad (13)$$

El valor del exponente δ, se utiliza para controlar el efecto de la actualización, en la sección de experimentos computacionales se discute como seleccionar dicho parámetro. Con este enfoque reactivo, los valores de α_i y λ_j con los que se han obtenido mejores soluciones tendrán mayor probabilidad de ser seleccionados en iteraciones subsecuentes.

El pseudocódigo de un método GRASP Reactivo en el cual se actualizan las probabilidades cada *U* iteraciones es presentado en el Algoritmo 3. Donde:

P_A : es la distribución de probabilidad de los valores de α.

P_L: es la distribución de probabilidad de los valores de λ.

f(s): función objetivo de la solución s.

Historia: es la memoria que registra el promedio de las distancias obtenidas con los diferentes valores de α y λ en las iteraciones anteriores.

```
Procedimiento GRASP_REACTIVO(Max_iteraciones,A,L)
  1.      Inicializar Historia, P_A, P_L
  2.      Para k=1 hasta Max_Iteraciones Haga
  3.            α←Selección Aleatoria de α(A,P_A)
  4.            λ←Selección Aleatoria de λ(L,P_L)
  5.            Solucion←Construccion_Aleatorizada(α,λ)
  6.            Solucion←Busqueda_Local(Solucion)
  7.            ActualizarMejorSolucion(Solucion,Mejor_Solucion)
  8.            Historia←Actualizacion_Historia(Historia,α,λ ,f(Solucion))
  9.            Si k mod U=0 entonces
  10.                 P_A←Actualizar_Probabilidades(A,Historia)
  11.                 P_L←Actualizar_Probabilidades(L,Historia)
  12.           Fin Si
  13.     Fin Para
  14.     Muestre  Mejor_Solucion
  15.     Fin GRASP_REACTIVO
```

Algoritmo 3. Pseudocódigo GRASP Reactivo

3.3. Búsqueda Local

Un algoritmo de búsqueda local explora repetidamente la vecindad de una solución, $N(s)$, en busca de una mejor. Cuando no se encuentra una solución que mejore la actual, la búsqueda se detiene, y se dice que la solución es localmente óptima. La búsqueda local juega un papel importante en GRASP ya que sirve para encontrar soluciones localmente óptimas en las regiones exploradas con la construcción aleatorizada. El Algoritmo 4 muestra un pseudocódigo para una búsqueda local.

```
Procedimiento Busqueda_Local(Solucion)
   1. Mientras Solucion no sea un optimo local Haga
   2.       Encuentre s'∈ N(Solucion) con f(s')<f(Solucion)
   3.       Solucion←s'
   4. Fin Mientras
   5.  Muestre Solucion
Fin Busqueda_Local
```

Algoritmo 4. Búsqueda Local. Adaptado de Resende & Ribeiro (2003)

3.4. GRASP/VND

En este trabajo se ha sustituido la búsqueda local simple utilizada en GRASP, con una variante de la búsqueda en vecindario variable VNS – (variable neighborhood search) conocida como VND (variable neighborhood descend) (Mladenovic & Hansen, 1997) (Hansen & Mladenovic, 2001).

VNS está basado en los siguientes principios:

- Un óptimo local con respecto a un vecindario no es necesariamente un óptimo local con respecto a otro vecindario.
- Un óptimo global es un óptimo local con respecto a todos los vecindarios.
- Para muchos problemas, los óptimos locales con respecto a uno o varios vecindarios están relativamente cerca unos a otros.

Estos tres principios se pueden usar de diferentes maneras, dando origen a las diferentes variantes de VNS.

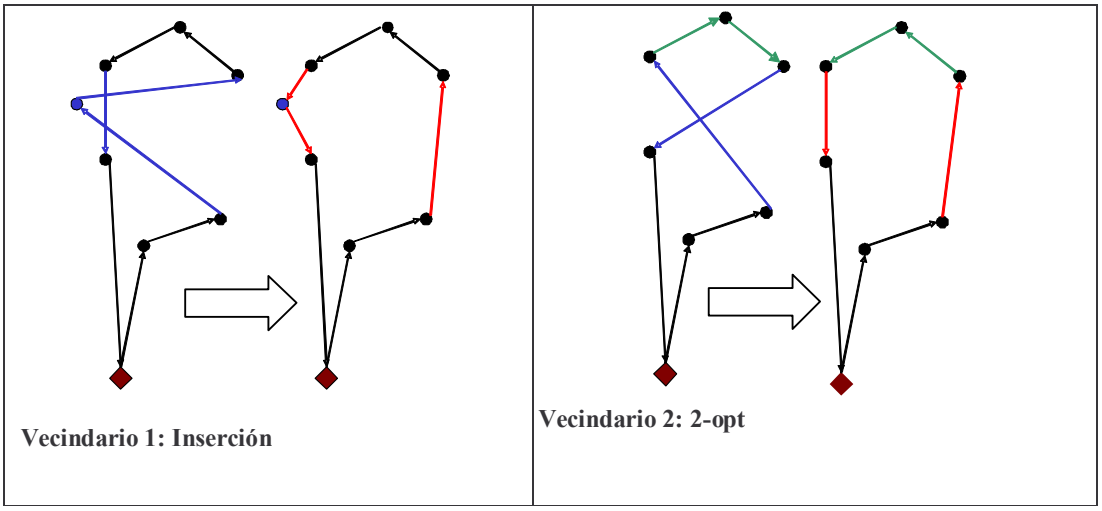
- Si se usan de manera determinística se tiene el método conocido como VND (Variable Neighborhood Descent).
- Si se usan estocásticamente se tiene el método RVNS (Reduced Variable Neighborhood Search)
- Si se usan en una combinación determinística- estocástica se tiene un VNS básico.

El esquema básico de VND se esboza en el Algoritmo 5. Donde $\mathbf{N_1}, \mathbf{N_2}, \ldots, \mathbf{N_k}$ son los vecindarios utilizados para mejorar la solución.

```
Procedimiento VND(Solucion)
   1. Seleccion el conjunto de vecindarios N1,N2,...,NL
   2. k=1
   3. Mientras k<L Haga
   4.       Encuentre s' en el vecindario Nk(Solucion)
   5.       Si f(s')<f(Solucion) Haga
   6.            Solucion←s'
   7.            k=1
   8.       Sino
   9.            k=k+1
   10.      Fin si
   11. Fin Mientras
   12.  Muestre Solucion
Fin VND
```

Algoritmo 5. Pseudocódigo VND. Adaptado de Hansen & Mladenovic (2001)

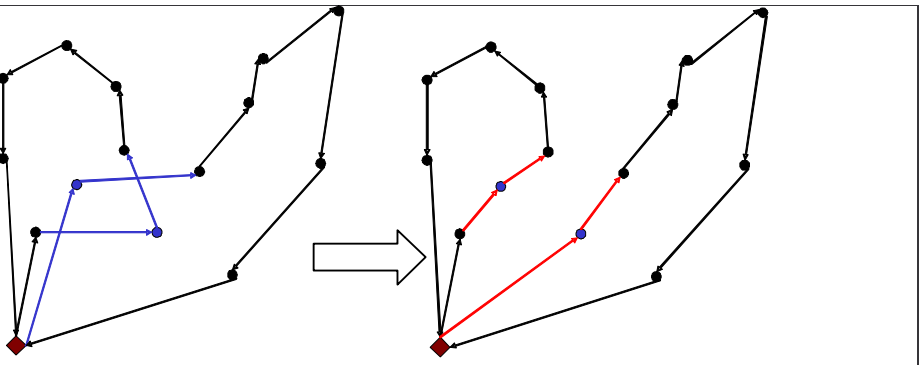
En este trabajo se decidió utilizar cinco vecindarios diferentes: *2-opt*, inserción, *exchange, relocate, y cross*. Los vecindarios se ilustran en la Figura 2 y la Figura 3. El color de los arcos tiene la siguiente convención: en azul se presentan los arcos que se eliminan, en rojo los arcos que se adicionan y en verde los arcos a los cuales se les cambia de sentido. La flecha indica el cambio de la solución (antes → después). Los métodos de búsqueda local para el VRP y otros problemas de ruteo se describen en detalle en Funke *et al*. (2005) y Kindervater & Savelsbergh (1997).



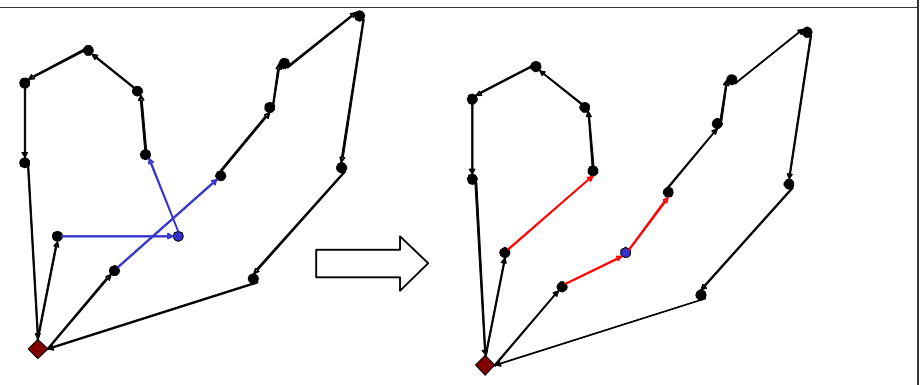
Vecindario 1: Inserción

Vecindario 2: 2-opt

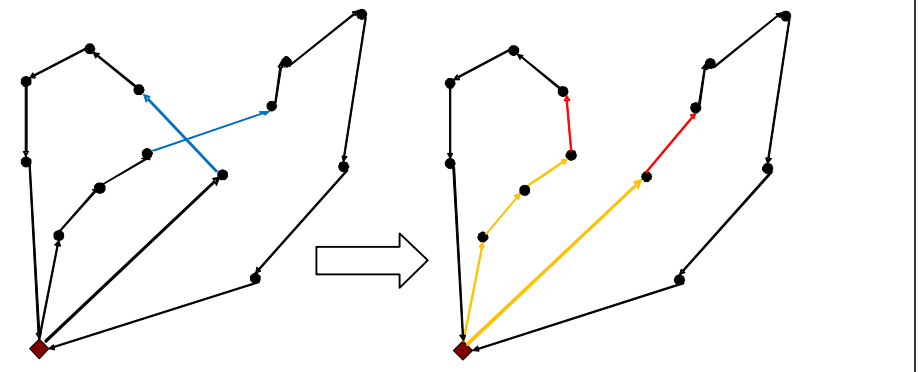
Figura 2. Vecindarios que operan sobre una ruta



Vecindario 3: Exchange



Vecindario 4: Relocate



Vecindario 5: Cross

Figura 3 Vecindarios que operan sobre dos rutas

Los vecindarios utilizados se exploran en orden de complejidad. Nótese que los vecindarios 1 y 2 operan sobre una sola ruta, los vecindarios 3 y 4 sobre un par de rutas, mientras que el vecindario 5 puede reducir el número de vehículos de la solución al fusionar dos rutas en una, cuando la capacidad lo

permite. El método GRASP/VND reactivo resultante se presenta en el algoritmo 6.

```
Procedimiento GRASP_VND_REACTIVO(Max_iteraciones,A,L)
   1.      Inicializar Historia, P_A, P_L
   2.      Para k=1 hasta Max_Iteraciones Haga
   3.            α←Selección Aleatoria de α(A,P_A)
   4.            λ←Selección Aleatoria de λ(L,P_L)
   5.            Solucion←Construccion_Aleatorizada(α,λ)
   6.            Solucion←VND(Solucion)
   7.            ActualizarMejorSolucion(Solucion,Mejor_Solucion)
   8.            Historia←Actualizacion_Historia(Historia,α,λ ,f(Solucion))
   9.            Si k mod U=0 entonces
   10.                 P_A←Actualizar_Probabilidades(A,Historia)
   11.                 P_L←Actualizar_Probabilidades(L,Historia)
   12.           Fin Si
   13.     Fin Para
   14.     Muestre  Mejor_Solucion
   15.     Fin GRASP_VND_REACTIVO
```

Algoritmo 6. Pseudocódigo Hibrido GRASP/VND

3.5. GRASP + memoria de frecuencia

Se implementó una última variante de GRASP, introducida por Ribeiro, *et al*. (2002), en la cual se utiliza una memoria de frecuencia con la cual se intenta "aprender" de las soluciones generadas previamente. Dicha variante utiliza una memoria basada en frecuencia, en la cual se cuenta el número de veces que cada elemento hace parte de los óptimos locales obtenidos en iteraciones previas. Esta información guía los procesos de perturbación de la fase constructiva. En esta variante de GRASP pueden utilizarse diferentes mecanismos orientados a la intensificación o la diversificación del proceso de búsqueda.

Utilizando un enfoque similar al propuesto en Ribeiro, *et al*., (2002), al método GRASP descrito en la sección anterior se le agregó la memoria de frecuencia, $M^t_{n+1\times n+1}$, (donde n es igual al número de clientes del problema y t es el contador del número de iteraciones), el elemento $m_{ij}(t)\in M^t$ es el número de veces que el arco (i, j) ha estado en las soluciones finales de las iteraciones previas a la iteración t.

El procedimiento original de perturbación de los ahorros se modifica según una de las siguientes alternativas:

- Si la opción seleccionada es diversificar la búsqueda, en regiones poco exploradas en las iteraciones previas:

$$s_{ij}^{r} = \frac{r \times s_{ij}}{1 + \frac{m_{ij}(t)}{t}} \qquad (14)$$

- Si la opción seleccionada es intensificar la búsqueda, en regiones ya exploradas:

$$s_{ij}^{r} = r \times s_{ij} \times \left(1 + \frac{m_{ij}(t)}{t}\right) \qquad (15)$$

- Si la alternativa seleccionada es una mezcla entre diversificación en intensificación (la cual en adelante se llamará estrategia de oscilación), se ejecuta la estrategia de diversificación en las iteraciones impares y la estrategia de intensificación en las iteraciones pares.

El pseudocódigo de un método híbrido GRASP/VND reactivo con memoria de frecuencia es el siguiente:

```
Procedimiento GRASP_VND_REACTIVO+MEMORIA(Max_iteraciones,A,L,M)
   1.      Inicializar Historia, P_A, P_L
   2.      Para k=1 hasta Max_Iteraciones Haga
   3.            α←Selección Aleatoria de α(A,P_A)
   4.            λ←Selección Aleatoria de λ(L,P_L)
   5.            Solucion←Construccion_Aleatorizada(M, α,λ)
   6.            Solucion←VND(Solucion)
   7.            ActualizarMejorSolucion(Solucion,Mejor_Solucion)
   8.            Historia←Actualizacion_Historia(Historia,α,λ ,f(Solucion))
   9.            M←Actualizacion_Memoria(M,Solucion)
   10.           Si k mod U=0 entonces
   11.                 P_A←Actualizar_Probabilidades(A,Historia)
   12.                 P_L←Actualizar_Probabilidades(L,Historia)
   13.           Fin Si
   14.     Fin Para
   15.     Muestre  Mejor_Solucion
Fin GRASP_VND_REACTIVO+MEMORIA
```

Algoritmo 7. Pseudocódigo Hibrido GRASP/VND Reactivo + Memoria

4. Experimento Computacional

Barr, *et al*. (1995) y Hooker (1995) resaltan lo importante que es contar con estudios comparativos de los diferentes métodos heurísticos aplicados a problemas de optimización combinatoria no sólo para reportar estrategias exitosas, sino también para medir el aporte de cada componente de un método heurístico, e incluso, para ilustrar estrategias que no tienen éxito. En este trabajo se realiza un experimento computacional cuyo objetivo es medir el desempeño de los diferentes métodos GRASP híbridos propuestos para el VRP; para ello se implementaron en *Visual Basic 6.0* las diferentes variantes de GRASP descritas en la sección 3. Como instancias de prueba se utilizaron los problemas clásicos de Christofides, *et al*. (1979) y las nuevas instancias de gran tamaño de Golden, *et al*. (1998).

Los problemas de prueba fueron descargados de Dorronsoro (2007); cada instancia se identifica utilizando las iniciales de los autores y el número de clientes, para los problemas CMT se utiliza además el mínimo de vehículos necesarios, por ejemplo el problema CMT100K8 corresponde a uno de los problemas de Christofides, *et al*. (1979) con 100 clientes y 8 vehículos, asimismo, GWKCh483 corresponde al problema de Golden, *et al*. (1998) con 483 clientes (en este caso se omite el dato de los vehículos porque no hay dos problemas con el mismo número de clientes así que no hay lugar a confusiones)

El experimento se realizó en un Intel Pentium IV a 3.2 GHz con 1GB bajo Windows XP Professional. Cada método se ejecuto una vez en cada problema, el tiempo de ejecución se reporta en segundos. Como métodos de comparación para medir el desempeño de los diferentes híbridos GRASP se tienen la implementación básica del método de los ahorros secuencial utilizada como subrutina de GRASP.

4.1. Componentes de GRASP

Como primer análisis, es importante medir el impacto del proceso de construcción aleatorizada (en adelante RCW- *Randomized Clarke & Wright*) en comparación con el algoritmo de los ahorros determinístico (en adelante CW). Los parámetros utilizados para controlar RCW ($\lambda = 0.1$; $\alpha = 0.2$) fueron escogidos después de realizar experimentos preliminares y observar con cuales valores se obtuvieron los mejores resultados en el GRASP reactivo.

Para cada problema se ejecutó RCW 5000 veces ($\text{Max\_Iteraciones} = 5000$), número suficiente como para observar una porción plana en la grafica de progreso de la búsqueda, en la cual no se encuentran mejores soluciones a pesar de realizar más iteraciones.

La Tabla 2 permite afirmar que la aleatorización por si sola es suficiente para obtener mejores resultados en los problemas CMT; en promedio se logra reducir el costo de las soluciones de CW en un 7.8%. La columna reducción se calculó usando la expresión (16).

$$Reducción = \frac{c(CW) - c(Método\ Propuesto)}{c(CW)} \qquad (16)$$

Donde $c(CW)$ es el costo obtenido con el método de los ahorros y $c(Método\ Propuesto)$ es el costo obtenido con el método propuesto (en este caso RCW). Valores positivos de reducción indican que el método propuesto obtiene una solución mejor, mientras que valores negativos indican que el método empeora con respecto a CW.

Tabla 2. Comparación CW y RCW en los problemas CMT

Problema	CW	RCW		
	Costo	Costo	Tiempo (s)	Reducción
CMT50	640.73	565.22	9.08	11.78%
CMT75	1.077.23	908.94	29.83	15.62%
CMT100K8	1.040.13	934.71	45.09	10.14%
CMT100K10	885.62	889.15	42.97	-0.40%
CMT120	1.261.59	1.188.75	55.25	5.77%
CMT150	1.271.13	1.210.20	110.06	4.79%
CMT200	1.622.57	1.507.42	234.19	7.10%
Promedio				7.8%

Por su parte, la Tabla 3 muestra como en los problemas GWKCh ocurre lo contrario, el costo con la construcción aleatorizada es mayor un 4.0% en promedio, cuando se compara con el obtenido con CW, es decir, la construcción aleatorizada empeora la calidad de las soluciones obtenidas.

Tabla 3 Comparación CW y RCW en los problemas GWKCh

Problema	CW	RCW		
	Costo	Costo	Tiempo (s)	Reducción
GWKCh240	800.77	785.47	425.61	1.91%
GWKCh252	937.71	967.55	534.33	-3.18%
GWKCh255	674.00	715.54	367.69	-6.16%
GWKCh300	1140.73	1112.84	804.45	2.45%
GWKCh320	1201.26	1251.00	996.80	-4.14%
GWKCh323	855.78	932.18	657.50	-8.93%
GWKCh360	1541.43	1526.86	1375.97	0.95%
GWKCh396	1488.02	1561.32	1721.20	-4.93%
GWKCh399	1047.26	1182.86	1082.50	-12.95%
GWKCh420	2060.77	2014.58	2143.86	2.24%
GWKCh480	1848.47	1923.00	2381.34	-4.03%
GWKCh483	1299.61	1451.61	1747.77	-11.70%
PROMEDIO				-4.0%

La Figura 4 muestra un crecimiento cuadrático del tiempo de ejecución de RCW con el tamaño del problema, el cual es medido usando el número de clientes. El tiempo de CW se omite ya que siempre fue menor a un segundo.

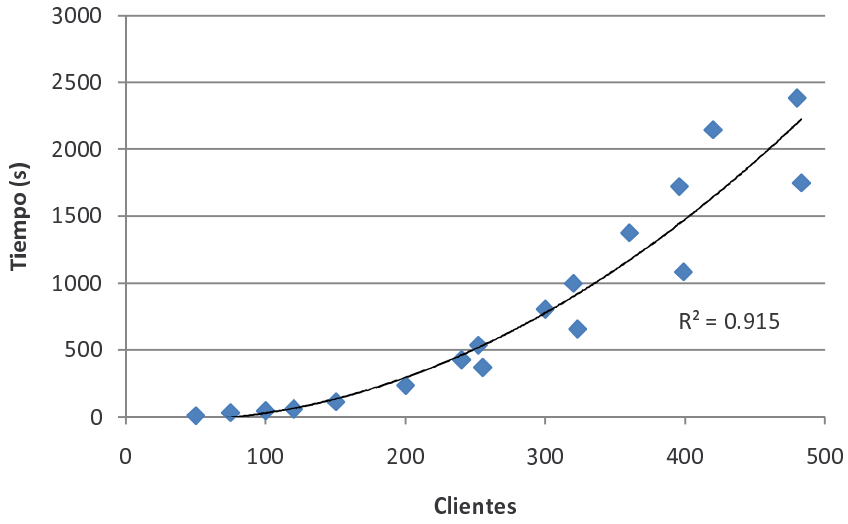


Figura 4 Tiempo de ejecución RCW vs. Número de clientes del problema

Igualmente, en la Figura 5 se llevo a cabo un reporte periódico del progreso de la búsqueda, cada 100 iteraciones se revisa cual es el costo de la mejor solución encontrada y de la solución construida en dicha iteración, la Figura 5 muestra la evolución de la búsqueda para el problema GWKCh240.

Como puede notarse, no se encuentra una solución mejor más allá de las 2700 iteraciones (aprox.). Además la calidad de la solución de cada iteración no mejora necesariamente por el hecho de tener más iteraciones, esto queda claro al ver que el rango en el cual se mueve el costo no disminuye cuando se tiene un número mayor de iteraciones. De cierta manera este fenómeno se debe a la falta de información que conecte una iteración con la siguiente.

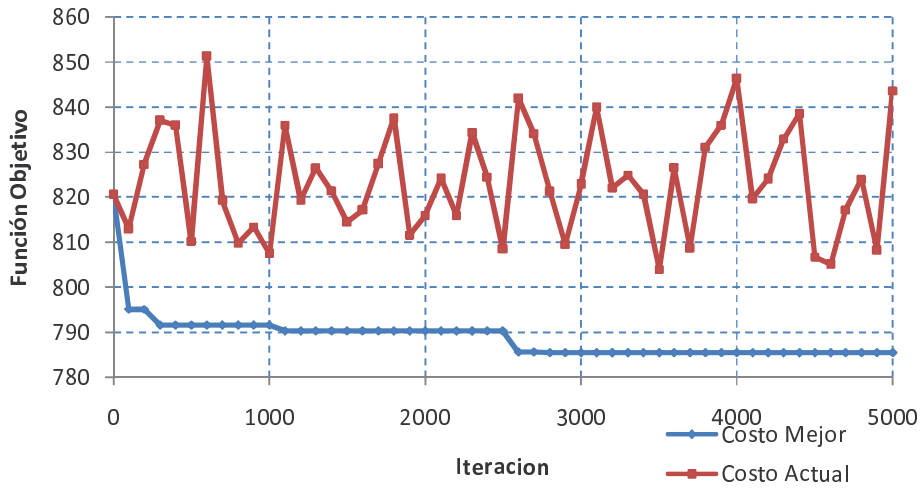


Figura 5 Evolución de la búsqueda (Problema GWKCh240)

El comportamiento observado en el problema GWkCh240 es bastante típico. Tal como se resume en la Tabla 4, en ningún problema se encontraron, con RCW, mejores soluciones después de las 4000 iteraciones, y en los problemas pequeños (CMT) después de las 2000 iteraciones.

Tabla 4 Iteración de la mejor solución encontrada con RCW

Problema	Mejor Solución encontrada en la Iteración	Problema	Iteración en la que se encontró la mejor solución
CMT50	1201	GWKCh300	3676
CMT75	504	GWKCh320	3519
CMT100K8	361	GWKCh323	1239
CMT100K10	330	GWKCh360	657
CMT120	1294	GWKCh396	3173
CMT150	1801	GWKCh399	1262
CMT200	3005	GWKCh420	1943
GWKCh240	2708	GWKCh480	3095
GWKCh252	3920	GWKCh483	1736
GWKCh255	3735	**Máximo**	3920

Después de analizar el aporte de la aleatorización, se incorporó VND como fase de mejora para llegar al híbrido GRASP/VND, cuyo desempeño en los problemas de prueba se resume en las columnas 7 y 8 de la Tabla 5. Como métodos de comparación se tienen CW, RCW y CW+VND, este último se obtiene al aplicar VND a la solución inicial encontrada con CW. Los parámetros α, λ y Max_Iteraciones, se mantuvieron en los valores utilizados para RCW. Nuevamente el método no aleatorizado tienen tiempos de

ejecución pequeños, en este caso CW+VND, con menos de cuatro segundos para cada problema.

Tabla 5 Resultados CW+VND y GRASP/VND

Problema	CW	RCW		CW+VND		GRASP/VND	
	Costo	Costo	Tiempo (s)	Costo	Tiempo (s)	Costo	Tiempo (s)
CMT50	640.73	565.22	9.08	594.35	0.02	547.10	36.47
CMT75	1077.23	908.94	29.83	943.33	0.03	857.70	122.44
CMT100K8	1040.13	934.71	45.09	925.37	0.13	843.98	271.97
CMT100K10	885.62	889.15	42.97	827.32	0.08	825.95	238.39
CMT120	1261.59	1188.75	55.25	1176.80	0.06	1050.35	216.27
CMT150	1271.13	1210.20	110.06	1174.52	0.33	1068.80	802.50
CMT200	1622.57	1507.42	234.19	1486.26	0.28	1374.66	1323.66
GWKCh240	800.77	785.47	425.61	781.09	0.47	745.58	2844.02
GWKCh252	937.71	967.55	534.33	931.31	0.31	926.68	2830.27
GWKCh255	674.00	715.54	367.69	657.64	0.81	646.70	2531.45
GWKCh300	1140.73	1112.84	804.45	1114.76	0.45	1064.55	2858.16
GWKCh320	1201.26	1251.00	996.80	1184.81	1.05	1181.79	5571.05
GWKCh323	855.78	932.18	657.50	837.07	1.05	825.21	4788.37
GWKCh360	1541.43	1526.86	1375.97	1523.67	1.42	1446.35	8440.08
GWKCh396	1488.02	1561.32	1721.20	1484.44	0.98	1473.54	10398.97
GWKCh399	1047.26	1182.86	1082.50	1036.24	1.47	1030.33	8938.25
GWKCh420	2060.77	2014.58	2143.86	2001.49	3.00	1933.36	11657.83
GWKCh480	1848.47	1923.00	2381.34	1788.85	3.78	1794.37	17958.86
GWKCh483	1299.61	1451.61	1747.77	1274.11	3.06	1251.56	16145.34

La Figura 6 y la Figura 7 permiten comparar el desempeño de los diferentes métodos, mostrando la reducción en la función objetivo obtenida con la incorporación de los elementos del método GRASP propuesto. En los problemas CMT la aleatorización del método de los ahorros (RCW) es casi tan buena como aplicar VND a la solución obtenida con CW, y cuando se utiliza GRASP/VND se obtiene una mejora todavía mayor. Por su parte, en los problemas de mayor tamaño el desempeño es mucho más parejo, sin embargo, en casi todos los casos la mejor solución se encuentra usando GRASP/VND (con excepción del problema GWKCh480)

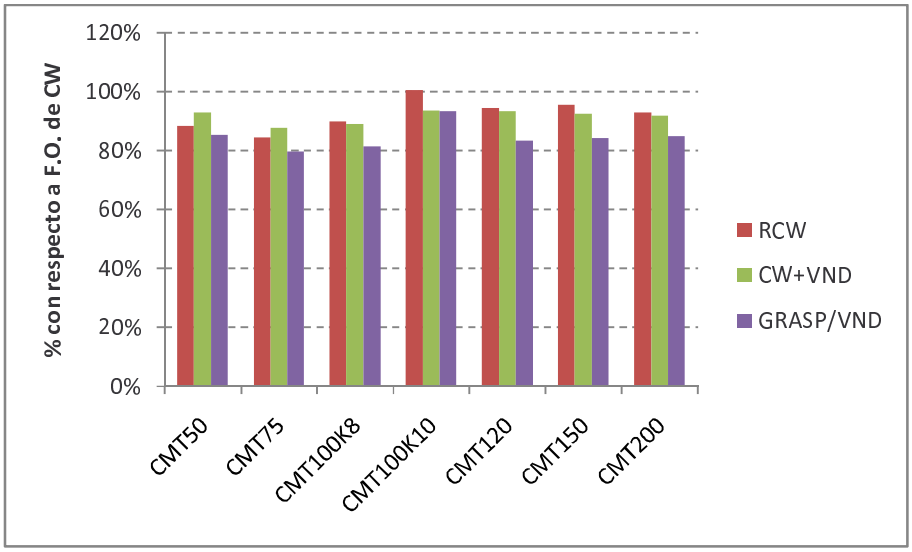


Figura 6 Costo obtenido con diferentes métodos en los problemas CMT

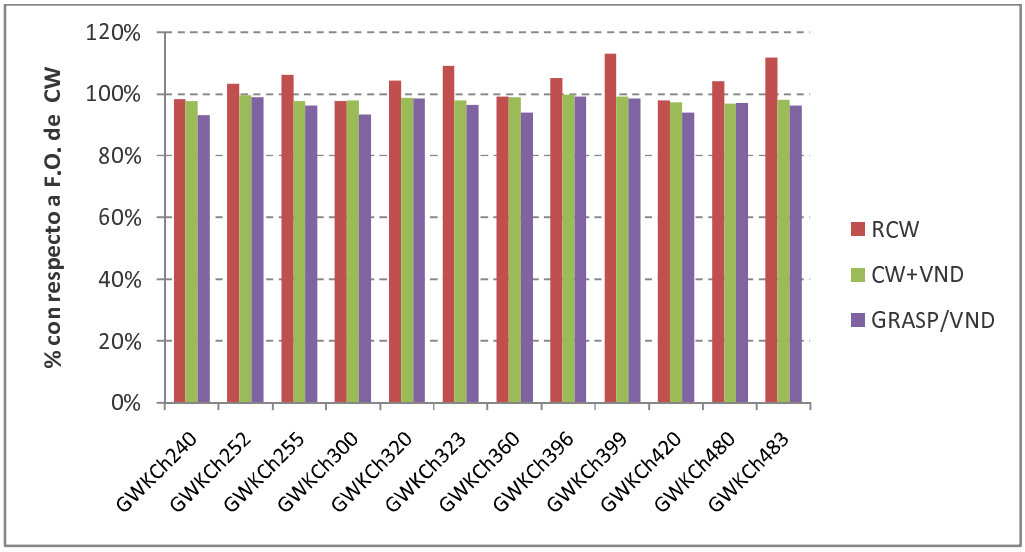


Figura 7 Costo obtenido con diferentes métodos en los problemas GWKCh

La calidad de la solución obtenida con GRASP/VND mejora notablemente en comparación con CW y RCW con reducciones de 4.3% y 7.7 % en promedio en la distancia total recorrida por los vehículos. Otro resultado importante es la mejora con respecto a CW+VND de 4.1%; lo cual muestra que la aleatorización de la fase constructiva es importante para que las búsquedas

locales encuentren buenas soluciones. La comparación se resume en la Tabla 6. Se usó nuevamente la expresión:

$$Reducción = \frac{c(Método\ Base) - c(Método\ Propuesto)}{c(Método\ Base)} \quad (17)$$

Tabla 6 Comparación GRASP/VND contra métodos base

Problema	GRASP/VND vs. CW	GRASP/VND vs. CW+VND	GRASP/VND vs. RCW
CMT50	7.2%	7.9%	3.2%
CMT75	12.4%	9.1%	5.6%
CMT100K8	11.0%	8.8%	9.7%
CMT100K10	6.6%	0.2%	7.1%
CMT120	6.7%	10.7%	11.6%
CMT150	7.6%	9.0%	11.7%
CMT200	8.4%	7.5%	8.8%
GWKCh240	2.5%	4.5%	5.1%
GWKCh252	0.7%	0.5%	4.2%
GWKCh255	2.4%	1.7%	9.6%
GWKCh300	2.3%	4.5%	4.3%
GWKCh320	1.4%	0.3%	5.5%
GWKCh323	2.2%	1.4%	11.5%
GWKCh360	1.2%	5.1%	5.3%
GWKCh396	0.2%	0.7%	5.6%
GWKCh399	1.1%	0.6%	12.9%
GWKCh420	2.9%	3.4%	4.0%
GWKCh480	3.2%	-0.3%	6.7%
GWKCh483	2.0%	1.8%	13.8%
Promedio	4.3%	4.1%	7.7%

Sin embargo, la mejora en la calidad de las soluciones viene acompañada de un aumento en el tiempo de ejecución bastante grande, incluso para el problema más pequeño el tiempo de ejecución se cuadruplica, y en promedio en tiempo de ejecución es seis veces mayor (última columna de la Tabla 7 y Figura 8). Este aumento en el tiempo de ejecución se ha medido usando un factor de tiempo expresado así:

$$Factor\ de\ tiempo = \frac{tiempo\ algoritmo\ con\ mejora}{tiempo\ algoritmo\ base} \quad (18)$$

Tabla 7 Comparación tiempos de ejecución RCW contra GRASP/VND

	Tiempo (s)		
PROBLEMA	RCW	GRASP/VND	Factor de tiempo contra RCW
CMT50	9.08	36.47	4.02
CMT75	29.83	122.44	4.10
CMT100K8	45.09	271.97	6.03
CMT100K10	42.97	238.39	5.55
CMT120	55.25	216.27	3.91
CMT150	110.06	802.50	7.29
CMT200	234.19	1323.66	5.65
GWKCh240	425.61	2844.02	6.68
GWKCh252	534.33	2830.27	5.30
GWKCh255	367.69	2531.45	6.88
GWKCh300	804.45	2858.16	3.55
GWKCh320	996.80	5571.05	5.59
GWKCh323	657.50	4788.37	7.28
GWKCh360	1375.97	8440.08	6.13
GWKCh396	1721.20	10398.97	6.04
GWKCh399	1082.50	8938.25	8.26
GWKCh420	2143.86	11657.83	5.44
GWKCh480	2381.34	17958.86	7.54
GWKCh483	1747.77	16145.34	9.24
Promedio			6.03

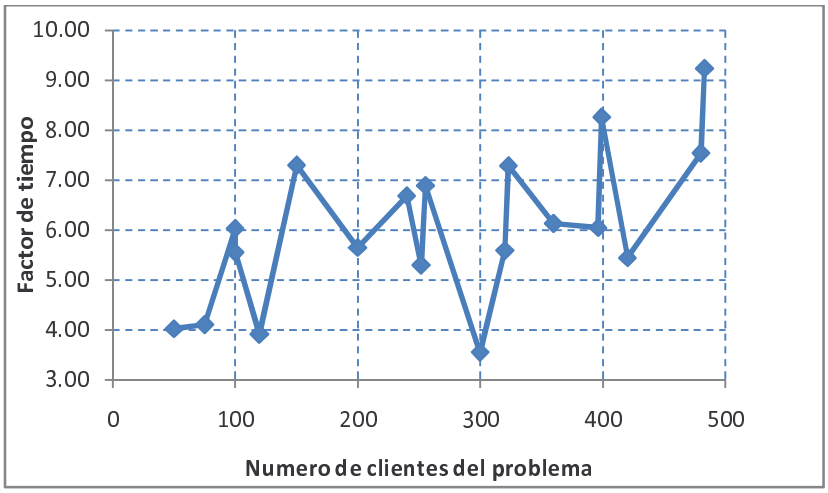


Figura 8 Tiempo de ejecución RCW vs. GRASP/VND

4.2. GRASP Reactivo

Paralelamente, se analizó el aporte del control reactivo, luego de correr algunos experimentos previos con el control reactivo se identificaron dos variantes para analizar, una en la cual los rangos de λ y α tomaban valores en un rango amplio y otra en la cual el rango es más pequeño, siempre con pasos de 0.1, en ambos casos en la expresión (12) se tomó $\delta = 2$, la Tabla 8 resume dichas variantes. Aunque originalmente se tienen cuatro posibilidades de parametrización de GRASP/VND, solamente se usaron las dos mencionadas.

Tabla 8. Parámetros del control reactivo

Parámetro	Rango pequeño	Rango amplio
λ	[0.1,0.5]	[0.1,1.5]
α	[0.1,0.5]	[0.1,1.0]

En la Tabla 9 se comparan las dos posibilidades para ver si una es mejor que la otra. Al comparar los dos métodos reactivos se encuentra que la calidad de las soluciones no es muy diferente (mejora de apenas 0.07% en favor del rango pequeño), pero si se observa que el método de rango pequeño es más rápido. Por lo tanto, en adelante se usará ese rango de parámetros para los demás experimentos y comparaciones.

Tabla 9 Resultados control reactivo rango amplio y rango pequeño

	GRASP/VND/REACTIVO Rango amplio (RA)		GRASP/VND/REACTIVO Rango pequeño (RP)		Factor de tiempo RA/RP	RA vs. RP
Problema	Costo	Tiempo (s)	Costo	Tiempo (s)		
CMT50	530.67	49.53	531.73	42.17	1.17	0.2%
CMT75	854.04	157.61	858.84	137.55	1.15	0.6%
CMT100K8	842.20	401.64	844.07	315.11	1.27	0.2%
CMT100K10	819.56	352.58	820.92	274.26	1.29	0.2%
CMT120	1055.81	366.19	1050.84	280.53	1.31	-0.5%
CMT150	1077.56	1209.72	1084.76	912.70	1.33	0.7%
CMT200	1387.87	1943.36	1387.42	1547.86	1.26	0.0%
GWKCh240	744.58	6459.39	745.59	4252.09	1.52	0.1%
GWKCh252	921.74	6047.03	921.28	4036.44	1.50	0.0%
GWKCh255	643.15	5748.20	640.93	3924.95	1.46	-0.3%
GWKCh300	1078.01	8167.39	1074.60	5115.48	1.60	-0.3%
GWKCh320	1177.67	12523.20	1176.60	8210.17	1.53	-0.1%
GWKCh323	825.91	11768.56	822.95	7890.53	1.49	-0.4%
GWKCh360	1454.83	20499.06	1460.48	13475.02	1.52	0.4%
GWKCh396	1470.04	23005.80	1468.30	15719.75	1.46	-0.1%
GWKCh399	1030.94	21677.86	1028.23	15052.81	1.44	-0.3%
GWKCh420	1952.18	29195.03	1952.29	19496.16	1.50	0.0%
GWKCh480	1794.73	44372.06	1782.61	28612.69	1.55	-0.7%
GWKCh483	1258.77	40245.91	1247.14	27228.44	1.48	-0.9%
Promedio					1.41	-0.07%

Seguidamente, se diseñó un experimento para medir el impacto del parámetro δ en el control reactivo, para ello se utilizó GRASP/VND/Reactivo con rango pequeño y tres valores diferentes de δ $(2, 2^3 \text{ y } 2^5)$, en todos los experimentos se utilizó una frecuencia de actualización de la probabilidad $U = 100$, lo cual significa que pasadas 100 iteraciones se realiza el procedimiento de actualización de probabilidad descrito en 3.3.

Con este experimento, se observó que no había un impacto importante de dicho parámetro en el desempeño del algoritmo (el control reactivo es poco sensible al valor de δ). Esta afirmación se sustenta en que la mejora promedio con respecto a CW para los tres métodos fue la misma (8.20%), y en muchos problemas los resultados fueron idénticos, tal como se aprecia en la Tabla 10.

Tabla 10 Impacto de δ en el desempeño del control reactivo

Problema	$\delta = 2$		$\delta = 2^3$		$\delta = 2^5$	
	Costo	Reducción vs. CW	Costo	Reducción vs. CW	Costo	Reducción vs. CW
CMT50	531.7275	17.01%	531.7275	17.01%	531.7275	17.01%
CMT75	858.8417	20.27%	858.8417	20.27%	858.8417	20.27%
CMT100K8	844.0742	18.85%	844.0742	18.85%	844.0742	18.85%
CMT100K10	820.9213	7.31%	820.9213	7.31%	820.9213	7.31%
CMT120	1050.839	16.71%	1050.839	16.71%	1050.839	16.71%
CMT150	1084.757	14.66%	1085.646	14.59%	1084.757	14.66%
CMT200	1387.416	14.49%	1391.001	14.27%	1387.416	14.49%
GWKCh240	745.5942	6.89%	745.5942	6.89%	745.5942	6.89%
GWKCh252	921.2825	1.75%	921.2825	1.75%	921.2825	1.75%
GWKCh255	640.9315	4.91%	640.9315	4.91%	640.9315	4.91%
GWKCh300	1074.601	5.80%	1070.701	6.14%	1074.601	5.80%
GWKCh320	1176.601	2.05%	1177.151	2.01%	1176.601	2.05%
GWKCh323	822.9543	3.84%	824.806	3.62%	822.9543	3.84%
GWKCh360	1460.477	5.25%	1451.751	5.82%	1460.477	5.25%
GWKCh396	1468.3	1.33%	1460.849	1.83%	1468.3	1.33%
GWKCh399	1028.234	1.82%	1027.46	1.89%	1028.234	1.82%
GWKCh420	1952.294	5.26%	1950.547	5.35%	1952.294	5.26%
GWKCh480	1782.605	3.56%	1782.605	3.56%	1782.605	3.56%
GWKCh483	1247.141	4.04%	1260.517	3.01%	1247.141	4.04%
Promedio		8.20%		8.20%		8.20%

Finalmente, es importante comparar el desempeño del control GRASP/VND Reactivo con el método GRASP/VND en el cual los valores de α y λ son fijos. Dicha comparación se presenta en la Tabla 11. La última columna de la Tabla 11 muestra como el método reactivo es ligeramente mejor (con una reducción promedio de la función objetivo, con respecto al método con parámetros fijos, de casi un 0.1%). Pero con tiempos de ejecución 44% mayores, que se deben a la carga computacional adicional que exige la estrategia reactiva y a la necesidad de GRASP reactivo de realizar algunas iteraciones con combinaciones de parámetros no muy buenas, en las que la búsqueda local converge lentamente.

Tabla 11 Comparación GRASP/VND/ Reactivo vs. GRASP/VND

Problema	GRASP/VND		GRASP/VND/REACTIVO		Factor de tiempo vs. GRASP/VND	Reducción vs. GRASP/VND
	Costo	Tiempo (s)	Costo	Tiempo (s)		
CMT50	547.10	36.47	531.73	42.17	1.16	2.8%
CMT75	857.70	122.44	858.84	137.55	1.12	-0.1%
CMT100K8	843.98	271.97	844.07	315.11	1.16	0.0%
CMT100K10	825.95	238.39	820.92	274.26	1.15	0.6%
CMT120	1050.35	216.27	1050.84	280.53	1.30	0.0%
CMT150	1068.80	802.50	1084.76	912.70	1.14	-1.5%
CMT200	1374.66	1323.66	1387.42	1547.86	1.17	-0.9%
GWKCh240	745.58	2844.02	745.59	4252.09	1.50	0.0%
GWKCh252	926.68	2830.27	921.28	4036.44	1.43	0.6%
GWKCh255	646.70	2531.45	640.93	3924.95	1.55	0.9%
GWKCh300	1064.55	2858.16	1074.60	5115.48	1.79	-0.9%
GWKCh320	1181.79	5571.05	1176.60	8210.17	1.47	0.4%
GWKCh323	825.21	4788.37	822.95	7890.53	1.65	0.3%
GWKCh360	1446.35	8440.08	1460.48	13475.02	1.60	-1.0%
GWKCh396	1473.54	10398.97	1468.30	15719.75	1.51	0.4%
GWKCh399	1030.33	8938.25	1028.23	15052.81	1.68	0.2%
GWKCh420	1933.36	11657.83	1952.29	19496.16	1.67	-1.0%
GWKCh480	1794.37	17958.86	1782.61	28612.69	1.59	0.7%
GWKCh483	1251.56	16145.34	1247.14	27228.44	1.69	0.4%
Promedio					1.44	0.09%

4.3. GRASP con memoria de frecuencia

Por último se analizó el efecto de la memoria de frecuencia, para ello se implementaron las tres estrategias descritas (intensificación, diversificación y oscilación) sobre el método GRASP/VND. Inicialmente no se utilizó el control reactivo para poder aislar el aporte de la memoria de frecuencia en el desempeño del método de solución; y se consideró la misma combinación de parámetros utilizada previamente para GRASP/VND, en la tabla 12 se compara el desempeño de las tres estrategias de implementación de la memoria de frecuencia contra CW. Las estrategias de oscilación e intensificación son las mejores en comparación con CW (reducción promedio

del 8.8% y del 8.49% respectivamente) y superan la estrategia de diversificación (reducción promedio del 7.8% contra CW).

	CW	Diversificación		Intensificación		Oscilación	
PROBLEMA	Costo	Costo	Reducción vs. CW	Costo	Reducción vs. CW	Costo	Reducción vs. CW
CMT50	640.73	530.57	17.2%	550.66	14.1%	529.09	17.4%
CMT75	1077.23	861.10	20.1%	869.24	19.3%	860.58	20.1%
CMT100K8	1040.13	844.87	18.8%	845.45	18.7%	843.56	18.9%
CMT100K10	885.62	819.56	7.5%	825.95	6.7%	820.92	7.3%
CMT120	1261.59	1054.05	16.5%	1053.83	16.5%	1053.39	16.5%
CMT150	1271.13	1079.31	15.1%	1082.56	14.8%	1083.17	14.8%
CMT200	1622.57	1388.07	14.5%	1377.02	15.1%	1384.39	14.7%
GWKCh240	800.77	746.72	6.7%	749.84	6.4%	746.91	6.7%
GWKCh252	937.71	932.07	0.6%	914.85	2.4%	919.89	1.9%
GWKCh255	674.00	649.71	3.6%	627.87	6.8%	632.58	6.1%
GWKCh300	1140.73	1069.20	6.3%	1069.20	6.3%	1069.20	6.3%
GWKCh320	1201.26	1182.50	1.6%	1174.11	2.3%	1177.71	2.0%
GWKCh323	855.78	830.98	2.9%	803.78	6.1%	804.68	6.0%
GWKCh360	1541.43	1463.78	5.0%	1455.62	5.6%	1452.12	5.8%
GWKCh396	1488.02	1473.43	1.0%	1461.53	1.8%	1453.75	2.3%
GWKCh399	1047.26	1035.80	1.1%	1007.38	3.8%	1015.36	3.0%
GWKCh420	2060.77	1947.74	5.5%	1945.64	5.6%	1938.33	5.9%
GWKCh480	1848.47	1801.74	2.5%	1788.34	3.3%	1782.49	3.6%
GWKCh483	1299.61	1268.53	2.4%	1224.81	5.8%	1231.69	5.2%
Promedio			7.8%		8.5%		8.7%

Tabla 12 Comparación estrategias de GRASP/Memoria de Frecuencia

También es importante comparar el desempeño de GRASP con memoria con respecto a GRASP/VND, en este caso sólo se comparó la estrategia de oscilación ya que es la mejor. La Tabla 13 muestra una reducción promedio del 0.6% que viene acompañada de poca carga computacional adicional pues el tiempo de ejecución solo crece en promedio un 13% (y en unos pocos casos disminuye). Es importante aclarar que no se ha incorporado el control reactivo, puesto que se quiere analizar aisladamente el aporte de la memoria basada en frecuencia.

Tabla 13 Comparación GRASP/Memoria contra GRASP/VND

Problema	GRASP/VND		GRASP/Memoria (Oscilación)		Reducción vs. GRASP/VND	Factor de tiempo vs. GRASP/VND
	Costo	Tiempo (s)	Costo	Tiempo (s)		
CMT50	547.10	36.47	529.09	37.01	3.3%	1.01
CMT75	857.70	122.44	860.58	112.17	-0.3%	0.92
CMT100K8	843.98	271.97	843.56	269.70	0.0%	0.99
CMT100K10	825.95	238.39	820.92	249.42	0.6%	1.05
CMT120	1050.35	216.27	1053.39	214.48	-0.3%	0.99
CMT150	1068.80	802.50	1083.17	749.20	-1.3%	0.93
CMT200	1374.66	1323.66	1384.39	1306.14	-0.7%	0.99
GWKCh240	745.58	2844.02	746.91	3416.47	-0.2%	1.20
GWKCh252	926.68	2830.27	919.89	3108.67	0.7%	1.10
GWKCh255	646.70	2531.45	632.58	2957.53	2.2%	1.17
GWKCh300	1064.55	2858.16	1069.20	4036.06	-0.4%	1.41
GWKCh320	1181.79	5571.05	1177.71	6269.33	0.3%	1.13
GWKCh323	825.21	4788.37	804.68	5645.25	2.5%	1.18
GWKCh360	1446.35	8440.08	1452.12	11187.45	-0.4%	1.33
GWKCh396	1473.54	10398.97	1453.75	11814.50	1.3%	1.14
GWKCh399	1030.33	8938.25	1015.36	10631.78	1.5%	1.19
GWKCh420	1933.36	11657.83	1938.33	16239.23	-0.3%	1.39
GWKCh480	1794.37	17958.86	1782.49	21040.58	0.7%	1.17
GWKCh483	1251.56	16145.34	1231.69	19176.66	1.6%	1.19
					0.6%	1.13

4.4. Comparación con resultados de la literatura

Por último, hace falta comparar el desempeño de un GRASP híbrido que incluye todas las estrategias analizadas (es decir, GRASP/VND con memoria de frecuencia y control reactivo) contra los resultados de la literatura. Nuevamente, se utilizaron los problemas de prueba CMT y GWKCh, pero esta vez se corrieron cinco réplicas con un máximo de 3000 iteraciones por réplica; y las comparaciones ya no se realizan contra CW sino contra los mejores métodos metaheurísticos reportados en la literatura (Cordeau *et al*., 2005).

Los resultados obtenidos en las cinco replicas (mejor solución, promedio y tiempo promedio) se reportan en la Tabla 14, la comparación reportada en la última columna de la tabla se calculó usando la mejor solución de las cinco

réplicas. Como puede observarse en la tabla, el desempeño del método GRASP hibrido propuesto es aceptable (en particular en los problemas pequeños y de tamaño intermedio (CMT)) con una función objetivo mayor en promedio un 2.2%, mientras que en los problemas de gran tamaño (GWKCh) se obtienen desviaciones un poco más grandes con respecto a las mejores soluciones reportadas en la literatura, entre 4.4% y 11.4%, y al parecer esta desviación es creciente con el tamaño del problema.

Tabla 14 Comparación GRASP hibrido contra resultados de la literatuwra

Problema	Mejor solución conocida	GRASP hibrido			
		Mejor 5 replicas	Promedio 5 replicas	Tiempo promedio (s)	GRASP híbrido vs. Mejor solución conocida
CMT50	524.61	524.61	533.32	23.21	0.0%
CMT75	835.26	855.07	859.55	71.82	-2.4%
CMT100K8	826.14	839.68	842.63	173.14	-1.6%
CMT100K10	819.56	820.92	820.92	155.39	-0.2%
CMT120	1042.11	1045.89	1051.08	158.43	-0.4%
CMT150	1028.42	1074.35	1080.13	507.77	-4.5%
CMT200	1291.29	1375.81	1384.70	871.20	-6.5%
GWKCh240	707.79	738.72	741.68	2543.18	-4.4%
GWKCh252	859.11	916.01	921.49	2233.40	-6.6%
GWKCh255	583.39	634.99	637.72	2357.01	-8.8%
GWKCh300	998.73	1063.88	1069.55	3100.32	-6.5%
GWKCh320	1081.31	1166.44	1172.27	4708.79	-7.9%
GWKCh323	742.03	813.51	814.91	6084.99	-9.6%
GWKCh360	1366.86	1441.84	1448.75	10614.95	-5.5%
GWKCh396	1345.23	1454.69	1461.73	11620.85	-8.1%
GWKCh399	919.45	1009.96	1017.62	9948.03	-9.8%
GWKCh420	1821.15	1941.94	1947.42	12307.25	-6.6%
GWKCh480	1622.69	1771.30	1778.38	16604.88	-9.2%
GWKCh483	1107.19	1233.38	1237.82	17035.02	-11.4%
Promedio					-5.8%

5. Conclusiones y trabajo futuro

En este trabajo se presentaron diferentes variantes de GRASP para la solución del VRP, en la variante más sencilla la etapa de construcción aleatorizada utiliza el algoritmo de los ahorros y la búsqueda local utiliza VND, este método permite obtener mejoras importantes con respecto al método de los ahorros (Clarke & Wright, 1964); con un metaheurístico sencillo y flexible. Además, en problemas pequeños la aleatorización es suficiente para mejorar el desempeño en comparación con la versión determinística del heurístico de los ahorros. La etapa de búsqueda local con VND si bien permite mejorar bastante las soluciones encontradas en la etapa constructiva, implica también, un tiempo de ejecución significativamente mayor.

Mejoras adicionales se logran al incluir un control reactivo en el esquema básico de GRASP, el cual permite encontrar los parámetros adaptativamente, y con el cual se elimina la necesidad de afinar los parámetros que controlan el metaheurístico. Igualmente, un mejor desempeño se logra agregando una memoria de frecuencia que utiliza la información de iteraciones anteriores para sesgar la selección de los elementos en la construcción aleatorizada. Esta mejora adicional, sin embargo, exige sacrificar parte de la sencillez característica del esquema básico de GRASP. Al comparar los resultados obtenidos con los mejores métodos reportados en la literatura se encuentra que el desempeño de GRASP es aceptable, con desviaciones no muy grandes con respecto a las mejores soluciones conocidas.

En el futuro, para lograr soluciones de mejor calidad, el método GRASP desarrollado puede utilizarse como generador de soluciones iníciales de metaheurísticos evolutivos (algoritmo genético, búsqueda dispersa o re-encadenamiento de trayectorias). Igualmente, una segunda posibilidad es la utilización de la memoria de frecuencia en la fase de búsqueda local para

obtener un hibrido GRASP/GLS (*Guided local search* (Voudouris, et al., 1998).

Finalmente, es importante resaltar que con el desarrollo de este trabajo se logró ampliar el conocimiento de GRASP cuando es aplicado al VRP, el aporte de cada uno de sus componentes y las posibilidades de mejora del esquema básico. Conocimiento y métodos que pueden utilizarse en el futuro para resolver otras variantes del VRP, otros problemas de optimización combinatoria y como componentes de otros metaheurísticos.

6. Bibliografía

[1] Baker, B., & Carreto, C. A. (2003). A visual interactive approach to vehicle routing. *Computers and Operations Research , 30* (3), 321-337.

[2] Baldacci, R., Toth, P., & Vigo, D. (2007). Recent advances in vehicle routing exact algorithms. *4OR , 5* (4), 269-298.

[3] Barr, R., Golden, B., Kelly, J., Rescende, M., & Stewart, W. (1995). Designing and reporting on computational experiments with heuristic methods. *Journal of Heuristics , 1* (1), 9-32.

[4] Christofides, N., Mingozzi, A., & Toth, P. (1979). The vehicle routing problem. En N. Christofides, A. Mingozzi, P. Toth, & C. Sandi, *Combinatorial Optimization* (págs. 315-338). Chichester UK: Wiley.

[5] Clarke, G., & Wright, J. (1964). Scheduling of vehicles from a central depot to a number of delivery points. *Operations Research , 12*, 568–581.

[6] Corberán, A., Fernández, E., Laguna, M., & Martí., R. (2002). Heuristic Solutions to the Problem of Routing School Buses with Multiple Objectives. *Journal of the Operational Research Society , 53*, 427-435.

[7] Cordeau, J. F., Gendreau, M. L., Potvin, J. I., & Semet, F. (2002). A guide to vehicle routing heuristics. *53*, págs. 512–533.

[8] Cordeau, J. F., Gendreau, M., Hertz, A., Laporte, G., & Sormany, J. S. (2005). New heuristics for the VRP. En A. Langevin, & D. Riopel, *Logistics systems: design and optimization* (págs. 279-297). New York: Springer.

[9] Cordeau, J.-F., Laporte, G., Savelsbergh, M., & Vigo, D. (2007). Vehicle Routing. En C. Barnhart, & G. Laporte, *Transportation, Handbooks in Operations Research and Management Science Vol.14* (págs. 367-428). Amsterdam: Elsevier.

[10] Dantzig, G. B., & Ramser, J. H. (1959). The Truck Dispatching Problem. *Management Science , 6* (1), 80-91.

[11] Dorronsoro, B. *The VRP Web*. Recuperado el 1 de Marzo de 2007 de http://neo.lcc.uma.es/radi-aeb/WebVRP/

[12] Faulin, J., Sarobe, P., & Simal, J. (2005). The DSS LOGDIS Optimizes Delivery Routes for FRILAC's Frozen Products. *Interfaces , 35* (3), 202–214.

[13] Feo, T., & Resende, M. (1989). A probabilistic heuristic for a computationally difficult set covering problem . *Operations Research Letters , 8*, 67-71.

[14] Festa, P., & Resende, M. G. (29 de 02 de 2004). *An annotated bibliography of GRASP*. Recuperado el 1 de febrero de 2007, de http://www.research.att.com/%7Emgcr/doc/gannbib.pdf

[15] Foster, B., & Ryan, D. (1976). An Integer Programming Approach to the Vehicle Scheduling Problem. *Operations Research, 27:367- , 27*, 367-384.

[16] Funke, B., Grünert, T., & Irnich., S. (2005). Local Search for Vehicle Routing and Scheduling Problems: Review and Conceptual Integration. *Journal of Heuristics , 11* (4), 267-306.

[17] Gendreau, M., & Potvin, J.-Y. (2005). Metaheuristics in Combinatorial Optimization. *Annals of Operations Research , 140* (1), 189 – 213.

[18] Gendreau, M., J.-Y. P., Bräysy, O., Hasle, G., & Løkketangen, A. (2008). Metaheuristics for the vehicle routing problem and its extension: A categorized bibliography. En B. Golden, S. Raghavan, & E. Wasil, *The Vehicle Routing Problem: Latest Advances and New Challenges*. New York: Springer.

[19] Gillett, B., & Miller, L. (1974). Heuristic Algorithm for the Vehicle-Dispatch Problem. *Operations Research , 22* (2), 340-349.

[20] Girard S., Renaud J. & Boctor F.F. A simple and efficient perturbation heuristic to solve the vehicle routing problema. Reporte No 2005-016 Faculté des sciences de l'administration ; Université Laval. - 2005.

[21] Golden, B., Assad, A., & Wasil, E. (2002). Routing Vehicles in the Real World: Applications in the Solid Waste,Beverage, Food, Dairy, and Newspaper Industries. En P. Toth, & D. Vigo, *The Vehicle Routing Problem* . Philadelphia: SIAM.

[22] Golden, B., Wasil, E., Kelly, J., & Chao, I.-M. (1998). The impact of metaheuristics on solving the vehicle routing problem: algorithms, problem sets, and computational results. En T. Crainic, & G. Laporte, *Fleet management and logistics* (págs. 33-56). Boston: Kluwer.

[23] Hall, R., & Partyka. J. (02 de 2008). *OR/MS Today vehicle routing survey*. Recuperado el 23 de mayo de 2008, de http://www.lionhrtpub.com/orms/orms-2-08/frvrss.html

[24] Hansen, P., & Mladenovic, N. (2001). Variable neighborhood search: Principles and Applications. *European Journal of Operational Research , 130*, 449–467.

[25] Hasle, G., & Kloster, O. (2007). Industrial Vehicle Routing. En G. Hasle, K.-A. Lie, & E. Quak, *Geometric Modelling, Numerical Simulation, and Optimization* (págs. 397 - 435). Berlin: Springer.

[26] Hjorring, C. A. (1995). *The vehicle routing problem and local search metaheuristics.* Doctoral Thesis, University of Auckland.

[27] Hooker, J. (1995). Testing Heuristics: We Have It All Wrong. *Journal of Heuristics , 1*, 33-42.

[28] Jang, W., Lim, H. H., Crowe, T. J., Raskin, G., & Perkins, T. E. (2006). The Missouri Lottery Optimizes Its Scheduling and Routing to Improve Efficiency and Balance. *Interfaces , 36*, 302-313.

[29] Kindervater, G., & Savelsbergh, M. (1997). Vehicle Routing: Handling Edge Exchanges. En E. Aarts, & J. Lenstra, *Local Search in Combinatorial Optimization* (págs. 337-360). Chichester: Wiley.

[30] Kontoravdis, G., & Bard, J. (1995). A GRASP for the vehicle routing problem with time windows. *ORSA Journal on Computing , 7* (1), 10–23.

[31] Kytöjoki, J., Nuortio, T., Bräysy, O., & Gendreau, M. (2007). An efficient variable neighborhood search heuristic for very large scale vehicle routing problems. *Computers & Operations Research , 34* (9), 2743-2757.

[32] Laporte, G. (2007). What you should know about the vehicle routing problem. *Naval Research Logistics , 54* (8), 811 - 819.

[33] Laporte, G. & Semet, F. (2002). Classical Heuristics for the Capacitated VRP. En D. Vigo & P. Toth, *The Vehicle Rouitng Problem* (págs. 109-126). Philadelphia: SIAM.

[34] Lenstra, J. K., & Rinnooy Kan, A. H. (1981). Complexity of vehicle routing and scheduling problems. *Networks , 11*, 221-228.

[35] Martí, R. (2003). Multi-Start Methods. En F. G. Kochenberger, *Handbook of Metaheuristics* (págs. 355-368). Kluwer Academic Publishers.

[36] Miller, D., & Weaver, J. (1997). An integrated spatial DSS for scheduling and routing home-health- care nurses. *Interfaces , 27*, 35-48.

[37] Mladenovic, N., & Hansen, P. (1997). Variable neighborhood search Computer.and Operations Research. *Computer & Operations Research , 24*, 1097-1100.

[38] Poot, A., Kant, G., & Wagelmans, A. (2002). A savings based method for real-life vehicle routing problems. *Journal of the Operational Research Society , 12* (1), 57-68.

[39] Prais, M., & Ribeiro, C. (2000). Reactive GRASP: An application to a matrix decomposition problem in TDMA traffic assignment. *INFORMS Journal on Computing , 12*, 164–176.

[40] Prins, C. Memetic Algorithms for Routing Problems. *Spring School on Vehicle Routing*. Recuperado el 23 de mayo de 2007, de: http://neumann.hec.ca/chairedistributique/en/school2006_en.shtml

[41] Resende, M. (2008). Metaheuristic hybridization with GRASP. En INFORMS, *Tutorials in Operations Research*.

[42] Resende, M., & Ribeiro, C. (2003). Greedy randomized adaptive search procedures. En F. Glover, & G. Kochenberger, *Handbook of Metaheuristics* (págs. 219-249). Kluwer Academic Publishers.

[43] Resende, M., & Gonzalez-Velarde, J. L. (2003). GRASP: Procedimientos de búsqueda miope aleatorizado y adaptativo . *Inteligencia Artificial , 2*, 61-76.

[44] Ribeiro, C., Uchoa, E., & Werneck, R. (2002). A hybrid GRASP with perturbations for the Steiner problem in graphs. *INFORMS Journal on Computing , 14*, 228–246.

[45] Snyder, L. (2003). *VRP Solver*. Recuperado el 12 de Junio de 2008, de http://www.lehigh.edu/~lvs2/software.html#vrp

[46] Toth, P., & Vigo, D. (2002). *The Vehicle Routing Problem.* Philadelphia: SIAM.

[47] Voudouris, C., & Tsang, E. (1998). Guided local search and its application to the traveling salesman problem. *European Journal of Operations Research , 113*, 80–119.

[48] Weigel, D., & Cao, B. (1999). Applying GIS and OR Techniques to Solve Sears Technician-Dispatching and Home Delivery Problems. *Interfaces , 29*, 112-130.