Universidad Simón Bolívar
Departamento de Computación y Tecnología de la Información
CI-56512- Diseño de Algoritmos II
Trimestre Abril-Julio 2012

Proyecto 2 Resolviendo el CVRP

1. Planteamiento del Problema

El *Vehicle Routing Problem* (VRP) es uno de los problemas de optimización combinatoria más estudiados. El *Capacited Vehicle Routing Problem* (CVRP) es la variante base del VRP. Se desea que implemente un software, basado en una metaheurística de trayectoria, que encuentre soluciones de alta calidad para el CVRP. La información básica sobre el VRP y sus variantes como el CVRP la puede obtener del libro de publicado por Toth y Vigo [9]. El capítulo 5 del libro [9], Laporte y Semet presentan las heurísticas clásicas para el CVRP [6]. Entre esas heurísticas están las *heurísticas de construccción de una solución* y las *heurísticas de mejora de soluciones*. En el capítulo 6 del mismo libro, Gendreau et al. [3] presentan las metaheurísticas utilizadas para resolver el CVRP. Otra referencia sobre los métodos clásicos y modernos para resolver el CVRP lo presenta Laporte et al. [5]. El artículo Cordeau et al. [2] trata sobre las nuevas heurísticas para el CVRP. El libro Golden et al. [4] presenta una la revisión más reciente sobre las metaheurísticas usadas para resolver el VRP y la mayoría de sus variantes como CVRP, en la sección *Metaheuristics for the Vehicle Routing Problem and Its Extensions*.

El objetivo del proyecto es que usted diseñe e implemente un algoritmo heurístico que resuelva el CVRP, que este basado en alguna de estas metaheurísticas: *Tabu Search*, *Variable neighborhood search*, *Iterated local search* o en *Greedy Randomized Adaptive Search Procedure*. La implementación de su programa debe hacerse usando alguno de estos tres lenguajes de programación: C, C++ ó JAVA. Las instancias con las que debe probar su algoritmo heurístico son las propuestas por Christofides, Mingozzi and Toth [1]. Estas instancias han sido ampliamente utilizadas para probar algoritmos que resuelven el CVRP y se les proporcionará en el material de apoyo. Las mejores soluciones encontradas a esas instancias las puede ver en la página web `http://neo.lcc.uma.es/radi-aeb/WebVRP/`, en la sección *Known Best Results*. El archivo `vrpinfo.txt`, que se encuentra junto con las instancias CVRP, explica el formato de los archivos. Puede ver que dos de los datos de las instancias son el *maximum route time* y el *drop time*. Se tiene que el *maximum route time* es el tiempo máximo que puede durar una ruta y *drop time* es el tiempo en que uno se demora en atender a un cliente. Lo que indican esos datos es que la distancia recorrida de una ruta más tiempo de servicio de cada uno de los cliente de esa ruta (*drop time*), debe ser menor al tiempo máximo permitido para la ruta (*maximum route time*).

Su programa puede mostrar la información que considere relevante por la salida estándar. Al finalizar su programa debe haber creado un archivo llamado `stat.nombreInstancia`, donde `nombreInstancia` es el nombre de la instancia que resolvió. En este archivo se almacenará la siguiente información, separadas por líneas.

- Distancia de la mejor solución
- Iteración de la mejor solución
- Número total de iteraciones hechas por el programa
- Tiempo en que fue encontrada la mejor solución, en segundos
- Tiempo total de la corrida del algoritmo, en segundos
- Número de rutas de la solución
- Se imprime cada una de las rutas separadas por una líneas. Los clientes de las rutas son separados por un espacio en blanco. Se tiene como convención que el depósito tiene el número cero (0) y los demás clientes se les asigna números naturales. Una ruta comienza y termina en el depósito, (por ejemplo, 0 7 1 3 0)

Además del programa que resuelve el CVRP, debe realizar un informe que contenga los siguientes secciones:

- *Portada*
- *Introducción:*
  1. Motivación del proyecto.
  2. Breve descripción del problema.
  3. Descripción del contenido del informe.
- *Algoritmo para el CVRP:* Debe describir de manera clara y precisa el algoritmo que diseño para resolver el CVRP. La idea es que cualquier profesional competente en ciencias de la computación que lea esta sección, debe ser capaz de implementar la heurística que usted desarrolló y de reproducir los resultados que usted obtuvo. Debido a esto la sección debería contener por lo menos:
  1. Las estructuras de datos usadas para representar el problema y otras que considere importantes
  2. La descripción de los principales algoritmos para resolver el problema. Preferiblemente debe mostrar el pseudocódigo de esos algoritmos
  3. Los parámetros que usa su algoritmo y la justificación de los mismos.
  4. Cualquier otra información que usted considere relevante para la implementación del algoritmo.

  Debe justificar y explicar el diseño de su solución.
- *Instrucciones de operación:* Descripción detallada de como compilar y correr su aplicación, así como el estado actual de la misma
- *Resultados Experimentales y Discusión:* (Ver indicaciones más adelante)
- *Conclusiones y Recomendaciones*
- *Referencias bibliográficas*

En cuanto a la sección de *Resultados Experimentales y Discusión* se desea hacer un estudio experimental que permita caracterizar el rendimiento de la solución algorítmica propuesta por usted.

En caso de que su solución sea un algoritmo probabilístico, debe presentar dos tablas de resultados experimentales. La primera tabla debe tener los siguientes datos:

- Nombre de la instancia
- Distancia promedio de 5 corridas de la heurística
- Porcentaje de desviación de la distancia promedio de la heurística, con respecto a la solución óptima
- Desviación estándar del valor promedio de la heurística
- Distancia de la mejor solución obtenida en las 5 corridas de la heurística
- Número de ocurrencias de la mejor solución en las 5 corridas de la heurística

La segunda tabla debe mostrar los siguientes resultados:

- Nombre de la instancia
- Distancia de la solución óptima

- Distancia de la mejor solución obtenida en las 5 corridas de la heurística
- Porcentaje de desviación de la mejor solución de la heurística con respecto a la solución óptima
- Tiempo promedio de las 5 corridas de la heurísticas, en segundos.

Si su solución para el CVRP es un algoritmo determinístico, debe hacer una tabla de resultados experimentales con los siguientes datos:

- Nombre de la instancia
- Distancia de la solución óptima
- Distancia de la mejor solución obtenida por su algoritmo
- Porcentaje de desviación de su mejor solución, con respecto a la solución óptima
- Tiempo de su mejor solución, en segundos

También puede incluir cualquier otra tabla o gráfico que considere relevante. El porcentaje de desviación de una solución de la heurística, con respecto a la solución óptima se calcula con la siguiente fórmula: $\frac{distanciaHeur - distanciaOpt}{distanciaOpt} * 100$. Las pruebas las puede realizar en cualquier computador. En el informe debe indicar las características del mismo: el modelo de procesador, la velocidad del reloj del procesador, la memoria RAM del sistema y el sistema de operación instalado. Su aplicación debe poder instalarse y ejecutarse en un equipo del LDC.

2. Sobre la entrega

Este proyecto es en equipos de máximo dos personas y tiene un valor de 30 % de la nota final. Los que quieran tener su nota el día viernes 15 de junio de 2012, deben hacer entrega del proyecto antes del día jueves 14 de junio de 2012 a las 3:30 pm. De lo contrario puede entregar el proyecto el día lunes 18 de junio hasta 12:00 pm. Debe entregar el informe del proyecto **impreso** y por email un archivo **.tar.gz** con el código del proyecto. En caso de no encontrar al encargado del curso en su oficina puede dejar el informe en su casillero en el Departamento de Computción o introducirlo debajo de la puerta de su oficina.

3. Consideraciones Finales

- Cualquier error que sea hallado en este enunciado, así como cualquier tipo de observación adicional sobre el proyecto, serán publicadas como fe de erratas en la página web del curso. Es responsabilidad de los alumnos revisar periódicamente la misma
- No debe haber copia, ni intercambio de información específica ni ayuda detallada entre los alumnos del curso. El incurrir en cualquiera de las acciones descritas anteriormente tendrá como consecuencia sanciones severas

Referencias

[1] Christofides, N. *Combinatorial optimization.* A Wiley-Interscience Publication. Wiley, 1979.

[2] Cordeau, J., Gendreau, M., Hertz, A., Laporte, G., and Sormany, J. New heuristics for the vehicle routing problem. *Logistics systems: design and optimization* (2005), 279–297.

[3] Gendreau, M., Laporte, G., and Potvin, J. Metaheuristics for the capacitated vrp. *The vehicle routing problem 9* (2002), 129–154.

[4] Golden, B., Raghavan, S., and Wasil, E. *The vehicle routing problem: latest advances and new challenges*, vol. 43. Springer Verlag, 2008.

[5] Laporte, G., Gendreau, M., Potvin, J., and Semet, F. Classical and modern heuristics for the vehicle routing problem. *International transactions in operational research 7*, 4-5 (2000), 285–300.

[6] Laporte, G., and Semet, F. Classical heuristics for the capacitated vrp. *The vehicle routing problem 9* (2002), 109–128.

[7] Lin, S. Computer solutions of the traveling salesman problem. *Bell Systems Technical Journal 44* (1965), 2245–2269.

[8] Osman, I. Meta-strategy simulated annealing and tabu search for vehicle routing problem. *Annals Operations Research 41* (1993).

[9] Toth, P., and Vigo, D. *The Vehicle Routing Problem*. Monographs on Discrete Mathematics and Applications. Society for Industrial and Applied Mathematics, 2002.

Anexos

A. Formulación del CVRP

El CVRP consiste en la entrega de mercancía desde un depósito a un conjunto de clientes usando una flota de vehículos idénticos. Cada cliente demanda cierta cantidad de mercancía. Los vehículos tienen una capacidad limitada. La solución consiste en construir rutas que comiencen y finalicen en el depósito tratando de minimizar la distancia. Las restricciones a las que esta sujeto CVRP son las siguientes:

- La demanda de todos los clientes deben ser satisfecha
- Un cliente solo puede ser atendido por un vehículo
- El número de vehículos involucrados en la solución no tiene límite
- Los vehículos tienen un tiempo máximo de llegada al depósito

Antes de mostrar la formulación del CVRP describiremos la notación a utilizar.

Descripción de parámetros:

S = solución válida, es decir, un conjunto de rutas
N = número de clientes desde 1 hasta N
C = capacidad del vehículo
0 = índice del depósito
$N + 1$ = índice del depósito
r = una ruta perteneciente a una solución S
l_{ij} = distancia entre las ciudades i y j, tal que $i, j = 0, 1, 2, \ldots, N + 1$
$d_{(i)}(r)$ = demanda del cliente i en la ruta r
$tmv(r)$ = tiempo máximo de viaje que le es permitido hacer a un vehículo
$st_i(r)$ = tiempo de servicio del cliente i en la ruta r (conocido como *service time*)
$n(s)$ = número de clientes en una ruta incluyendo el depósito

Descripción de las variables:

$I_{ij}(S) = 1$ si el arco ij pertenece a la solución S, 0 de lo contrario
$at_i(r)$ = tiempo de arribo del cliente i en la ruta r (conocido como *arrival time*)
$tv_i(r)$ = tiempo de viaje del cliente i en la ruta r

$cv_i(r)$ = carga del vehículo i en la ruta r

A continuación se presenta el problema CVRP.

Se desea minimizar:

$$g(S) = \sum_{i \neq j} I_{ij}(S) \cdot l_{ij} \quad i, j = 0, 1, 2, \ldots, N+1 \tag{1}$$

sujeto a:

$$tv_{(0)}(r) = 0 \qquad r \in S \tag{2}$$

$$at_{(i)}(r) = tv_{(i-1)}(r) + l_{(i-1,i)}(r) \quad i = 0, 1, 2, \ldots, n(r) \quad r \in S \tag{3}$$

$$tv_{(i)}(r) = at_{(i)}(r) + st_{(i)}(r) \quad i = 0, 1, 2, \ldots, n(r) \quad r \in S \tag{4}$$

$$tv_{(i)}(r) - tmv(r) \leq 0 \quad i = 0, 1, 2, \ldots, n(r) \quad r \in S \tag{5}$$

$$cv_{(i)}(r) = \sum_i d_{(i)}(r) \qquad d_{(i)}(r) \geq 0 \qquad i = 0, 1, 2, \ldots, n(r) \quad r \in S \tag{6}$$

$$cv_{(i)}(r) - C \leq 0 \quad i = 0, 1, 2, \ldots, n(r) \quad r \in S \tag{7}$$

Tenemos que la restricciones que van desde la (2) hasta la (5) indican que el tiempo de viaje de una ruta debe ser menor que el tiempo máximo de viaje permitido a un vehículo. Las ecuaciones (6) y (7) describen que la demanda de los clientes que pertenecen a una ruta, no debe sobrepasar la capacidad del vehículo.

B. Búsqueda Local entre rutas

Después de obtener una solución factible se puede de mejorar su calidad por medio de un algoritmo de búsqueda local. Es posible realizar un algorimto búsqueda local entre rutas haciendo uso del método llamado λ-*interchange* que puede implementarse con los operadores *swap* y *move*. Estos dos operadores fueron propuestos por Osman [8]. A continuación se explicará en primer termino en que consiste el mecanismo de generación λ-*interchange*, el cual da como resultados los operadores *move* y *swap* para finalmente describir el algoritmo de búsqueda local λ-*interchange* que los aplica.

B.1. Mecanismo de Generación λ-*interchange*

Dado una posible solución para VRP representada por $S = \{R_1, .., R_p, .., R_q, .., R_k\}$, donde R_k es una ruta que es atendida por un vehículo k. Un λ-*interchange* entre 2 pares de rutas consiste en un reemplazo de dos subconjuntos de rutas que pertenecen a la solución inicial de un problema VRP. Sea $S_1 \subseteq R_p$ de tamaño $|S_1| \leq \lambda$ y sea $S_2 \subseteq R_p$ de tamaño $|S_2| \leq \lambda$ se construyen dos nuevas rutas de la forma $R'_p = (R_p - S_1) \cup S_2$ y la ruta $R'_q = (R_q - S_2) \cup S_1$, y teniendo como resultado el nuevo conjunto de soluciones $S' = \{R_1, .., R'_p, .., R'_q, .., R_k\}$. Se deriva que el conjunto formado por todos los $\{S'\}$ es generado por el método λ-*interchange* para un λ dado. En este enunciado solo se considera $\lambda = 1$. El mecanismo 1-*interchange* usa dos procesos para generar conjuntos de soluciones que llamamos operador *move* y *swap*.

B.1.1. Operador *move*

Este operador hace que un cliente se cambia de un ruta a otra. En la figura 1 tenemos un ejemplo de como trabaja este operador. Una característica importante de *move* es que es posible eliminar rutas si alguna de ellas posee un solo cliente en su haber.

B.1.2. Operador *swap*

Este operador intercambia cada cliente de una ruta, con todos los otros clientes de las otras rutas. Un ejemplo del operador lo podemos observar en la figura 2

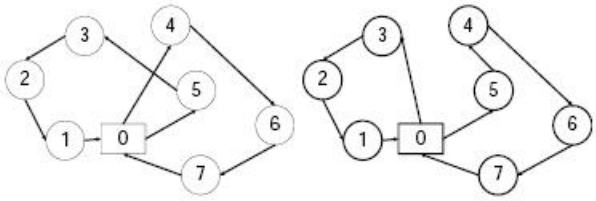

Figura 1: Ejemplo del operador *move*, el cliente 5 pasa de una ruta a otra

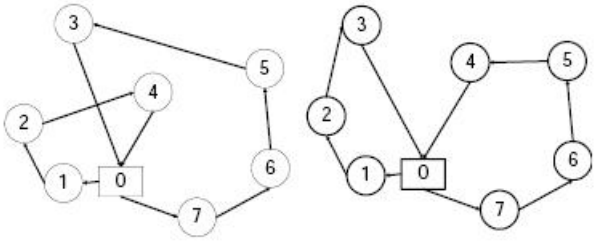

Figura 2: Ejemplo del operador *swap*, se intercambian los clientes 3 y 4 entre las rutas

B.2. Estrategia de selección

Se consideran dos estrategias de selección para escoger una solución de mejor calidad cuando se aplica la búsqueda local a partir de una solución inicial.

1. *First-Best*: Se acepta la primera solución que mejore la solución de partida
2. *Global Best*: Se comparan todas las soluciones que mejoren la solución inicial y se escoge la de menor costo. Tiene como principal desventaja el esfuerzo computacional que demanda

B.3. El algoritmo λ-*interchange*

Este algoritmo es el método de búsqueda local. Se comienza con una solución S obtenida por el paso de *generación de soluciones*. Luego se intenta mejorar S usando el mecanismo 1-*interchange*, el cual genera una nueva solución S'. La nueva solución S' es seleccionada usando el criterio *First-Best* o *Global Best*. El algoritmo continúa hasta que se encuentra con un mínimo local. Una solución S la consideramos un mínimo local si solo si $\text{Costo}(S) \leq \text{Costo}(S')\ \forall\ S' \in$ *conjunto de soluciones factibles*. En la figura 3 se presenta el algoritmo.

C. Búsqueda local intraruta

Cada una de las rutas de una solución factible del CVRP puede verse con una solución factible de un TSP. Por lo tanto es posible mejorar la distancia de la ruta haciendo uso de cualquier algoritmo de mejora de una solución TSP. Por ejemplo puede hacer uso de los médodos 2-opt ó 3-opt propuesto por Lin [7] para TSP. El método *k-opt* consiste en la sustitución de k-arcos, por otro conjunto de k-arcos de una ruta que forma parte de la solución. En la figura 4 se muestra un ejemplo del 2-opt y en la figura 5 tenemos el ejemplo de 3-opt en donde hay de 2 posibles intercambios.

```
Procedimiento búsqueda-local-interchange(S: Solución)
1 Inicio
2       S' := aplicar-operador(S) //Puede ser el operador swap o move
3       △ := Costo(S') - Costo(S)
4       Si (△ < 0 ) entonces
5               S := S'
6               IR A línea 2
7       Fin Si
8       Si se cumple un ciclo de búsqueda sin mejorar solución entonces
9               retornar S
10      De lo contrario
11              IR A línea 2
12      Fin Si
13 Fin
```

Figura 3: Algoritmo de búsqueda local λ-*interchange*

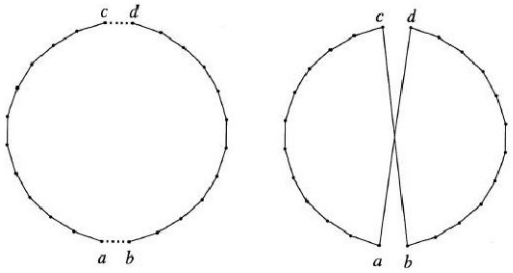

Figura 4: Ejemplo de 2-opt

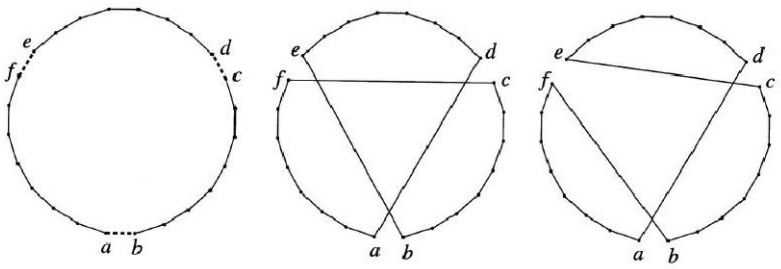

Figura 5: Dos combinaciones posibles después de aplicar 3-opt