Ruteo de Vehículos

Dpto. Ingeniería Industrial, Universidad de Chile

IN47B, Ingeniería de Operaciones

19 de abril de 2007

Contenidos

Definición

- Uno o más vehículos deben, desde una (o varias) bodega(s), visitar n clientes para luego volver a su punto de salida.
- Cada cliente debe ser visitado sólo una vez, y la distancia (o costo) total debe ser lo más corta posible.
- c_{ij} representa la distancia (o costo) entre el cliente i y el j.

Nota:

Caso particular con distancias euclidianas, y sólo un vehículo corresponde al problema más estudiado en Optimización Combinatorial, es decir, el TSP.

Observaciones

- Los únicos algoritmos exactos conocidos para resolver estos problemas tienen un tiempo de resolución que crece exponencialmente con el número de clientes n.
- Por esta razón, los esfuerzos se han destinado a desarrollar algoritmos heurísticos que permitan encontrar buenas soluciones.
- Los algoritmos desarrollados consideran desde ideas muy simples a ideas muy sofisticadas, en la práctica, las mejores heurísticas permiten tener soluciones muy cercanas a la óptima.

Datos

- Parámetros:
  - n Número de clientes.
  - c_{ij} Costo visitar cleinte j después del cliente i.
  - K Número de vehículos disponibles.
  - k_o Capacidad de cada vehículo.
  - d_i Demanda del cliente i.
- Variables:
  - x_{ijk} 1 si vehículo k visita j después de i, 0 en cualquier otro caso.
- Función Objetivo:

$$Z = \sum_{k=1}^{K} \sum_{i=1}^{n} \sum_{j=1, j\neq i}^{n} x_{ijk} c_{ij}.$$

Restricciones

- Entrar a cliente una vez.

$$\sum_{k=1,i=0:i\neq j}^{K,n} x_{ijk} = 1 \quad \forall j = 1,\ldots,n \tag{1}$$

- Salir de cliente una vez.

$$\sum_{k=1,i=0:i\neq j}^{K,n} x_{jik} = 1 \quad \forall j = 1,\ldots,n \tag{2}$$

- Eliminación de sub-circuitos.

$$\sum_{i,j\in S,k=1}^{K} x_{ijk} \leq |S| - 1 \quad \forall S \subset \{1,\ldots,n\} \tag{3}$$

Restricciones

- El vehículo que entra es el que sale.

$$\sum_{i=0,i\neq j}^{n} x_{ijk} - \sum_{i=0,i\neq j}^{n} x_{jik} = 0 \quad \forall j = 1,\ldots,n, k = 1,\ldots,K \tag{4}$$

- Cada vehículo sale del origen a lo más una vez.

$$\sum_{i=1}^{n} x_{0ik} \leq 1 \quad \forall k = 1,\ldots,K \tag{5}$$

- Capacidad de los vehículos

$$\sum_{i=1,j=1,i\neq j}^{n,n} x_{ijk} d_j \leq k_o \quad \forall k = 1,\ldots,K \tag{6}$$

Algunas variaciones comunes

- Ventanas de tiempo.
- Múltiples vueltas.
- Pickup and delivery.
- Distintos productos.
- Tipos de vehículos.
- Tiempo Real.

Límite de las heurísticas

- Si $\mathcal{P} \neq \mathcal{NP}$, ninguna heurística polynomial para el TSP puede garantizar $A(I)/OPT(I) \leq 2^n$ para todas las instancias I [SG76].
- Si $\mathcal{P} \neq \mathcal{NP}$, existe $\varepsilon > 0$ tal que ninguna heurística polynomial para el TSP puede garantizar $A(I)/OPT(I) \leq 1 + \varepsilon$ para todas las instancias I satisfaciendo la desigualdad triangular [ALM$^+$92].
- Existe un algoritmo A que, dado cualquier instancia euclidiana de TSP, y una constante $\varepsilon > 0$, se ejecuta en tiempo $n^{\mathcal{O}(1/\varepsilon)}$ y garantiza $A(I)/OPT(I) < 1 + \varepsilon$ [Aro96].

Vecino más próximo (NN)

- Iniciación $k = 1$, Escoger una ciudad inicial aleatoreamente $i_k \in \{1, \ldots, n\}$.
- Loop Mientras $k \neq n$, avanzar $k = k + 1$, y escoger $i_k \in \{1, \ldots, n\} \setminus \{i_1, \ldots, i_{k-1}\}$ que minimice $c(i_{k-1}, i_k)$.
- Salida Retornar el tour $(i_1, \ldots, i_n)$.
- Observaciones:
  - Ejecución es $\mathcal{O}(n^2)$.
  - Garantía $NN(I)/OPT(I) \leq \frac{1}{2}(\lfloor \log_2(n) \rfloor + 1)$.
  - Existen instancias tal que $NN(I)/OPT(I) \approx \Theta(\log(n))$.

Heurística Golosa (GR)

- Iniciación Ordenar los arcos $e \in V \times V$ en costo creciente $e_1, \ldots, e_M$, y asignar $m = k = 0$.
- Loop Mientras $k \neq n$, avanzar $k = k + 1$ y $m = m + 1$, Mientras $\{e_1, \ldots, e_{k-1}\} \cup \{e_m\}$ no formen parte de algún tour, avanzar $m = m + 1$. Asignar $e_k = e_m$.
- Salida Retornar tour descrito por $\{e_1, \ldots, e_n\}$.
- Observaciones:
  - Ejecución es $\mathcal{O}(n^2 \log(n))$.
  - Garantía $NN(I)/OPT(I) \leq \frac{1}{2}(\lfloor \log_2(n) \rfloor + 1)$.
  - Existen instancias tal que $NN(I)/OPT(I) \approx \Theta(\log(n)/3\log(\log(n)))$.

Clarke and Wright I

- Inicialización Crear n rutas $(0,i)(i,0)$.
- Calculo de ahorros Computar $s_{ij} = c_{i0} + c_{0j} - c_{ij}$ y ordenar de mayor a menor.
- Buscar pegado factible Dado el mejor ahorro s_{ij}, determinar si i,j están en rutas distintas, si su unión respeta capacidad, y si los arcos $(i,0)$ y $(0,j)$ son usados. Si es asi, unir ambas rutas eliminando $(i,0)$ y $(0,j)$ y agregando el arco (i,j), si no, descartar el ahorro.
- Condición de término Repetir paso 3 mientras existan ahorros factibles.
- Observaciones:

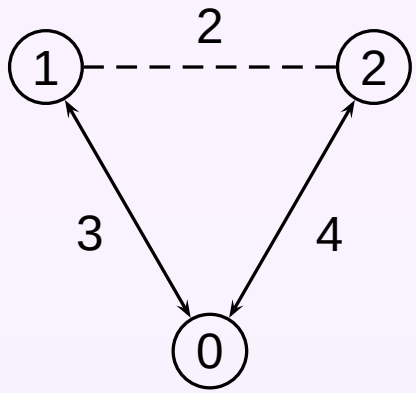
Clarke and Wright II

- Ejecución es $\mathcal{O}(n^2 \log(n))$.
- Garantía $NN(I)/OPT(I) \leq \lceil \log_2(n) \rceil + 1$.
- Existen instancias tal que $NN(I)/OPT(I) \approx \Theta(\log(n)/3\log(\log(n)))$.

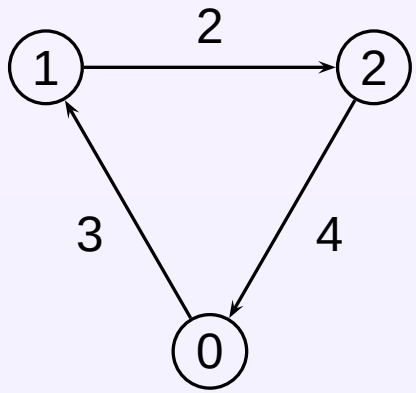
Clarke and Wright

Ejemplo

- Costo es 6 + 8 = 14
- $s_{12} = 3 + 4 - 2 = 5$.



- Costo es 3 + 2 + 4 = 9



Christofides (CHR)

- Paso 1 Construir un arbol T de peso mínimo en $G = (V, E)$, donde $V = \{1, \ldots, n\}$ y $E = V \times V$, note que $c(T) \leq c^*$.
- Paso 2 Construir un matching M entre los arcos de grado impar en T, note que $c(M) \leq \frac{1}{2}c^*$.
- Paso 3 Note que $M \cup T$ es conexo y euleriano (grado par), entonces existe $S \subset M \cup T$ tour en G.
- Observaciones:
  - Ejecución es $\mathcal{O}(n^3)$.
  - Garantía $NN(I)/OPT(I) \leq \frac{3}{2}$.
  - Existen instancias tal que $NN(I)/OPT(I) \approx \Theta(\frac{3}{2})$.

Método de dos Fases
Sweep Algorithm

- Paso 1 Escoger un vehículo no utilizado k.
- Paso 2 Escoger un cliente no servido con *ángulo* mínimo, y agregar clientes al vehículo k siempre que la capacidad sea respetada, recorriendo en ángulos el plano.
  Si quedan clientes sin asignar, volver al Paso 1.
- Paso 3 Optimizar la ruta de cada vehículo en forma separada, ya sea exactamente o heurísiticamente (El problema de cada vehículo es un TSP).

Comparación de heurísticas

N	10^2	10^3	10^4	10^5	10^6
Instancias aleatoreas euclidianas (SEP GAP)					
CHR	9.5	9.7	9.9	9.9	-
CW	9.2	11.3	11.9	12.1	12.2
GR	19.5	17.0	16.6	14.9	14.2
NN	25.6	26.0	24.3	23.6	23.3
Instancias aleatoreas (SEP GAP)					
GR	100	170	250	-	-
NN	130	240	360	-	-
CW	270	980	3200	-	-

K-opt I

- Si $\mathcal{P} \neq \mathcal{NP}$, ninguna heurística que cada paso tome tiempo polynomial, satisface $A(I)/OPT(i) \leq C$ para cualquier constante C, incluso si el número de movimientos es exponencial.
- Incluso si $\mathcal{P} = \mathcal{NP}$, ninguna heurística con vecindades polynomiales que no dependa de I, puede encontrar el óptimo de I.
- $2 - opt(I)/OPT(I) \geq \frac{1}{4}\sqrt{n}$ para instancias con desigualdad triangular.
- $3 - opt(I)/OPT(I) \geq \frac{1}{4}\sqrt[6]{n}$ para instancias con desigualdad triangular.

K-opt II

- $k - opt(I)/OPT(I) \geq \frac{1}{4}\sqrt[2k]{n}$ para instancias con desigualdad triangular.
- $k - opt(I)/OPT(I) \approx \mathcal{O}(\log(n))$ para instancias euclidianas.
- Existen instancias euclidianas donde $k - opt(I)/OPT(I) \approx \Theta(\log(n)/\log(\log(n)))$.

Comparación de heurísticas

N	10^2	10^3	10^4	10^5	10^6
Instancias aleatoreas euclidianas (SEP GAP)					
GR	19.5	17.0	16.6	14.9	14.2
CW	9.2	11.3	11.9	12.1	12.2
CHR	9.5	9.7	9.9	9.9	-
2-Opt	4.5	4.9	5.0	4.9	4.9
3-Opt	2.5	3.1	3.0	3.0	3.0
Instancias aleatoreas (SEP GAP)					
GR	100	170	250	-	-
2-Opt	34	70	125	-	-
3-Opt	10	33	63	-	-

Conceptos Básicos

- Dado $(P)\max cx : x \in S_1, Cx \leq d$ podemos definir $(P_\lambda)\max cx + \lambda(d - Cx) : x \in S_1$ para $\lambda \geq 0$.
- Note que $z_P^* \leq z_{P_\lambda}^*$.
- Puede demostrarse que $z_P^* = \min_{\lambda} z_{P_\lambda}^*$.

Relajación Lagrangeana para Ruteo

$$\min \sum_{k=1}^{K} \sum_{i=1}^{n} \sum_{j=1, j\neq i}^{n} x_{ijk} c_{ij}$$

$$\sum_{k=1, i=0: i\neq j}^{K, n} x_{ijk} = 1 \quad \forall j = 1, \ldots, n \tag{1}$$

$$\sum_{k=1, i=0: i\neq j}^{K, n} x_{jik} = 1 \quad \forall j = 1, \ldots, n \tag{2}$$

$$\sum_{i,j \in S} x_{ijk} \leq |S| - 1 \quad \forall S \subset \{1, \ldots, n\}, \forall k = 1, \ldots, K \tag{3}$$

$$\sum_{i=0, i\neq j}^{n} x_{ijk} - \sum_{i=0, i\neq j}^{n} x_{jik} = 0 \quad \forall j = 1, \ldots, n, k = 1, \ldots, K \tag{4}$$

$$\sum_{i=1}^{n} x_{0ik} \leq 1 \quad \forall k = 1, \ldots, K \tag{5}$$

$$\sum_{i=1, j=1, i\neq j}^{n, n} x_{ijk} d_j \leq k_o \quad \forall k = 1, \ldots, K \tag{6}$$

Bibliografía I

- S. Arora, C. Lund, R. Motwani, M. Sudan, and M. Szegedy, *Proof verification and hardness of approximation problems*, Proceedings 33rd Annual Sympossium on Fundations of Computer Science (Los Alamitos, CA), IEEE Computer Society Press, 1992, pp. 12–23.
- S. Arora, *Polynomial time approximation schemes for euclidian TSP and other geometric problems*, Proceedinfs 37th Annual Sympossium on Foundations of Computer Science (Los Alamitos, CA), IEEE Computer Society Press, 1996, pp. 2–11.

Bibliografía II


S. Shani and T. Gonzalez, *P-complete approximation problems*, Journal of the Association for Computing Machinery **23** (1976), 555–565.