Ruteo de Vehículos

Dpto. Ingeniería Industrial, Universidad de Chile

IN47B, Ingeniería de Operaciones

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 cliente j después de cliente i.
  - K Número de vehículos disponibles.
  - K_o Capacidad de cada vehículo.
  - d_j Demanda cliente j.
- Variables:
  - x_{ijk} 1 si vehículo k visita j después de i, 0 en cualquier otro caso.
- Función objetivo:

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

Restricciones

- Entrar a cliente una vez:

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

- Salir de cliente una vez:

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

- Eliminación de sub-circuitos:

$$\sum_{k=1,i,j\in S}^{K} x_{jik} \leq |S| - 1 \quad \forall \emptyset \neq S \subsetneq N \tag{3}$$

Restricciones

- El vehículo que entra es el que sale:

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

- Cada veículo sale del origen a lo más una vez:

$$\sum_{i\in N} x_{0ik} \leq 1 \quad \forall k \in K \tag{5}$$

- Capacidad de los vehículos:

$$\sum_{i\in N, j\in N\setminus\{i\}} x_{ijk} d_j \leq k_o \quad \forall k \in K \tag{6}$$

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)

$k \leftarrow 1$, $T = \emptyset$.
Escoger aleatoreamente $i_k \in N$, $T \leftarrow T \cup \{i_k\}$.
while $k \neq n$ **do**
    $k \leftarrow k + 1$, escoger $i_k \in N \setminus T$, $T \leftarrow T \cup \{i_k\}$ que minimice $c(i_{k-1}, i_k)$.
RETURN 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 I tal que $NN(I)/OPT(I) \approx \Theta(\log(n))$.

Heurística golosa (GR)

Ordenar $e \in E$ tal que $c_{e_i} \leq c_{e_{i+1}}$, $m \leftarrow k \leftarrow 0$, $T \leftarrow \emptyset$.
while $k \neq n$ **do**
  $k \leftarrow k + 1$, $m \leftarrow m + 1$
  while $T \cup \{e_m\}$ tiene sub-ciclos **do**
    $m \leftarrow m + 1$.
  $T \leftarrow T \cup \{e_m\}$.
RETURN $(e_1, \ldots, e_n)$.

Observaciones:

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

Clarke and Wright (CW)- Savings

1: $\mathcal{T} \leftarrow \{T_i = (0,i)(i,0)\}$, computar $s_{ij} = c_{i0} + c_{0j} - c_{ij}$, ordenar s de mayor a menor, $m \leftarrow 0$.
2: **while** $m \neq n^2$, $m \leftarrow m + 1$ **do**
3: **while** $s_m = s_{ij}$, i, j en misma ruta o $d(T_i + T_j) > k_o$ o ruta T_i no termina en i o ruta T_j no empieza en j **do**
4: $m \leftarrow m + 1$.
5: $\mathcal{T} \leftarrow \mathcal{T} \setminus \{T_i, T_j\} \cup \{T_i + T_j\}$.
6: RETURN Conjunto rutas $\mathcal{T}$.

Observaciones:

- Ejecución es $\mathcal{O}(n^2 \log(n))$.
- Garantía $CW(I)/OPT(I) \leq \lceil \log_2(n) \rceil + 1$.
- $\exists I: \; CW(I)/OPT(I) \approx \Theta(\log(n)/3 \log(\log(n)))$.

Christofides (CHR)

1: $T \leftarrow$ árbol peso mínimo en $G = (V, E)$ $c(T) \leq c^*$.
2: $M \leftarrow$ matching peso mínimo entre nodos grado impar en T $c(M) \leq \frac{1}{2}c^*$.
3: $M \cup T$ es un grafo euleriano y conexo, entonces existe un tour $S \subset M \cup T$, que satisface $c(S) \leq \frac{3}{2}c^*$
4: RETURN S.

Observaciones

- Ejecución es $\mathcal{O}(n^3)$.
- Garantía $CHR(I)/OPT(I) \leq \frac{3}{2}$.
- Existen instancias tal que $CHR(I)/OPT(I) \approx \Theta(\frac{3}{2})$.

Métodos de dos Fases:

Sweep Algorithm:

1: $k \leftarrow 0, \mathcal{T} \leftarrow \emptyset$.
2: **while** existan clientes sin visitar **do**
3: $k \leftarrow k + 1$.
4: Escoger un cliente no servido con ángulo mínimo, y agregar clientes al vehículo k hasta saturar capacidad, recorriendo el plano en ángulos.
5: $\mathcal{T} \leftarrow \mathcal{T} \cup \{ruta_k\}$.
6: Optimizar cada ruta en $\mathcal{T}$.
7: RETURN $\mathcal{T}$.

Comparación de Heurísticas

Instancias aleatoreas euclidianas (SEP GAP)

N	10^2	10^3	10^4	10^5	10^6
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)

N	10^2	10^3	10^4	10^5	10^6
GR	100	170	250	-	-
NN	130	240	360	-	-
CW	270	980	3200	-	-

K-opt

- 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

- $k - opt(I)/OPT(I) \geq \frac{1}{4} \sqrt[2k]{n}$ 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

Instancias aleatoreas euclidianas (SEP GAP)

N	10^2	10^3	10^4	10^5	10^6
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)

N	10^2	10^3	10^4	10^5	10^6
GR	100	170	250	-	-
2-Opt	34	70	125	-	-
3-Opt	10	33	63	-	-

Algunas conclusiones:

- Las mejores heurísticas para TSP son basadas en *local search*.
- Si existe algo de tiempo disponible (después de una solución factible), las mejores opciones son 2-Opt, 3-Opt y Link-Kernighan (y sus variantes).
- De momento, ninguna heurística basada en Tabu-Search, Simulated-Annealing o Neural-Networks se acercan al desempeño de estas heurísticas (tiempo de ejecución).
- Si hay más tiempo disponible, Itertated LK, y LKH son las mejores opciones.
- En términos de cotas, algunas meta-heurísticas obtienen resultados similares.
- Estos resultados son para el TSP.

Bibliografía I

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