Programación dinámica: Problema del viajante

Teoría de Algoritmos I (75.29 / 95.06)

Ing. Víctor Daniel Podberezski
vpodberezski@fi.uba.ar

Problema del viajante de comercio (TSP)

Sea

Un conjunto de n ciudades “C”

Un conjunto de rutas con costo de tránsito

Existe una ruta que une cada par de ciudades

El costo de cada ruta puede ser simétrico o asimétrico (diferente valor ida y vuelta)

Queremos

Obtener el circuito de menor costo

que inicia y finalice en una ciudad inicial

y pase por el resto de las ciudades 1 y solo 1 vez

Ejemplo

Contamos con 5 ciudades

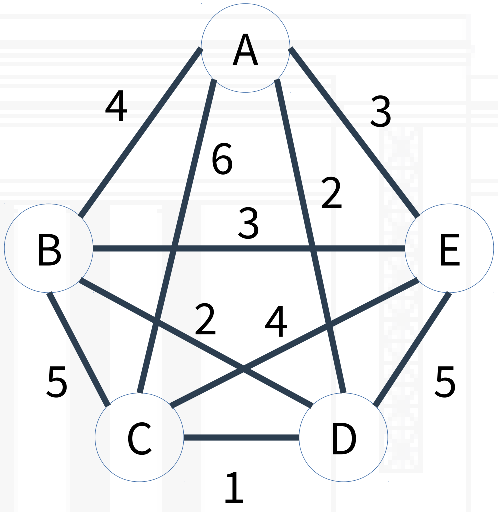
- El costo por camino es simétrico
- Partimos de A

Posibles ciclos:

- A – B – C – D – E – A → Costo: 18
- A – C – B – D – E – A → Costo: 21
- A – D – B – C – E – A → Costo: 16
- . . .

Ciclo de menor costo:

- A – D – C – B – E – A → Costo: 14
- A – D – C – E – B – A → Costo: 14



Solución por fuerza bruta

Tenemos que calcular todos los ciclos posibles.

Con n ciudades todas interconectadas (grafo completo)

Existen (n-1)! Ciclos de longitud n-1

Para cada ciclos calcular su costo

O(n)

Y quedarnos con el mínimo

Complejidad

$O(n^*(n-1)!) \rightarrow O(n!)$

Algoritmo Bellman–Held–Karp

Propuesto forma independiente en 1962

"Dynamic programming treatment of the travelling salesman problem" por Richard Bellman

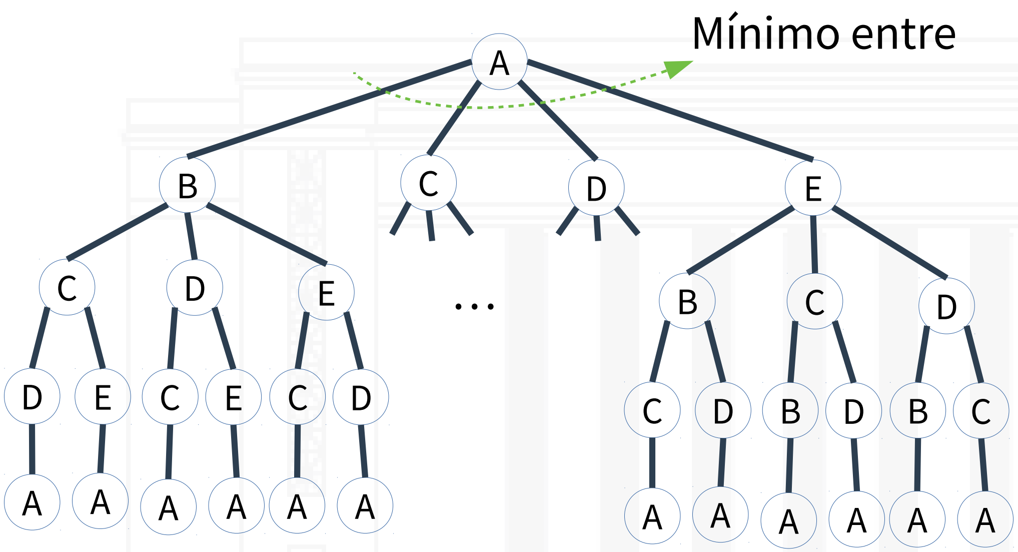
http://akira.ruc.dk/~keld/teaching/algoritmedesign_f08/Artikler/05/Bellman61.pdf

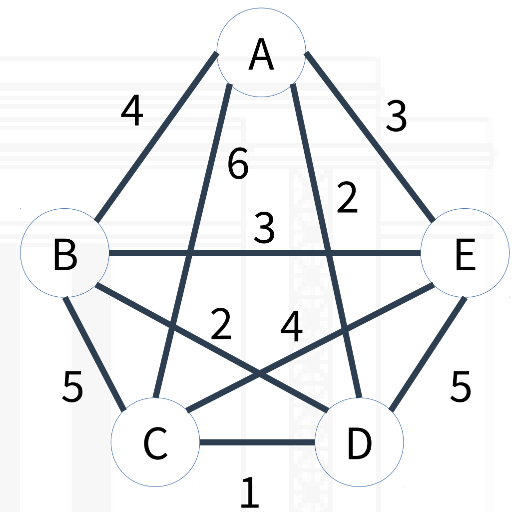
"A dynamic programming approach to sequencing problems" por Michael Held y Richard M. Karp"

Utiliza programación dinámica

Análisis

Podemos descomponer el problema como





Son (n-1)! hojas

… pero algunas ramas se ven repetidas

Análisis

El óptimo de problema

Se puede descomponer como el mínimo entre los subproblemas menores.

$$OPT(i,\{S\}) = min_{j\in\{S\}}\, w(i,j) + OPT(j,\{S-j\})$$

$$OPT(i,\emptyset) = w(i,\text{start}), \qquad \text{start: ciudad inicial}$$

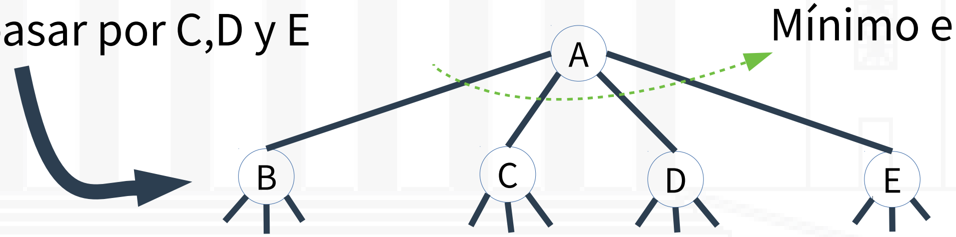
Con

{S} subset de ciudades

i ciudad de partida

Si voy por “B”, luego tengo que pasar por C,D y E

Mínimo entre



En el ejemplo:

Inicialmente parto de “A”

OPT(A,{B,C,D,E}) = min [w(A,B) + OPT(B,{C,D,E}) , w(A,C) + OPT(C,{B,D,E}) , w(A,D) + OPT(D,{B,C,E}) , w(A,E) + OPT(E,{B,C,D})]

Análisis (cont.)

Todos los subproblemas que tenemos que calcular:

OPT('B',{ C,D ,E}) , OPT('C',{ B, E, D }) , OPT('D',{ B, C, E }, OPT('E',{ B, C, D }),

OPT('B',{ C,D }) , OPT('B',{ C,E }) , OPT('B',{ D, E }),
OPT('C',{ B,D }) , OPT('C',{ B,E }) , OPT('C', { D, E }),
OPT('D',{ B,C }) , OPT('D',{ B,E }) , OPT('D',{ C, E }),
OPT('E',{ B,C }) , OPT('E',{ B,D }) , OPT('E',{ C, D }),

OPT('B',{ C }) , OPT('B',{ D }) , OPT('B',{ E }),
OPT('C',{ B }) , OPT('C',{ B }) , OPT('C',{ E }),
OPT('D',{ B }) , OPT('D',{ C }) , OPT('D',{ E }),
OPT('E',{ B }) , OPT('E',{ C }) , OPT('E',{ D }),

OPT('B', ∅) , OPT('C', ∅), OPT('D', ∅),OPT('E', ∅)

Para las n-1 ciudades excepto la ciudad de inicio (y fin)

Combinaciones de ciudades tomadas de diferente cantidad (según nivel subproblema)

Análisis (cont.)

Todos los subproblemas que tenemos que calcular:

Combinación de 3 ciudades

OPT('B',{ C,D ,E}) , OPT('C',{ B, E, D }) , OPT('D',{ B, C, E }, OPT('E',{ B, C, D }),

$$\binom{3}{3} * 4$$

Tomadas de a 3

$+$

OPT('B',{ C,D }) , OPT('B',{ C,E }) , OPT('B',{ D, E }),
OPT('C',{ B,D }) , OPT('C',{ B,E }) , OPT('C', { D, E }),
OPT('D',{ B,C }) , OPT('D',{ B,E }) , OPT('D',{ C, E }),
OPT('E',{ B,C }) , OPT('E',{ B,D }) , OPT('E',{ C, D }),

$$\binom{3}{2} * 4$$

$+$

OPT('B',{ C }) , OPT('B',{ D }) , OPT('B',{ E }),
OPT('C',{ B }) , OPT('C',{ B }) , OPT('C',{ E }),
OPT('D',{ B }) , OPT('D',{ C }) , OPT('D',{ E }),
OPT('E',{ B }) , OPT('E',{ C }) , OPT('E',{ D }),

$$\binom{3}{1} * 4$$

$+$

$$\binom{3}{0} * 4$$

OPT('B', ∅) , OPT('C', ∅), OPT('D', ∅),OPT('E', ∅)

Análisis (cont.)

Cantidad total de subproblemas:

Para n ciudades

$$\sum_{k=0}^{n-2} \binom{n-2}{k} * (n-1)$$

Que podemos expresar como:

$$(n-1) * \sum_{k=0}^{n-2} \binom{n-2}{k}$$

Y utilizando teorema del binomio con x=y=1

$$(n-1) * \sum_{k=0}^{n-2} \binom{n-2}{k} 1^{n-2-k} 1^{k} \quad = (n-1) * (1+1)^{(n-2)}$$

Teorema del binomio:

$$(x+y)^n = \sum_{k=0}^{n} \binom{n}{k} x^{n-k} y^k$$

Análisis (cont.)

Cantidad total de subproblemas:

$$=(n-1)*(2)^{(n-2)}$$

No es polinomial

… pero es mejor que exponencial para n grandes

Cada subproblema debe buscar el mínimo

Utilizando sus subproblemas

La comparación se ejecuta en tiempo lineal

Solución iterativa

Llamamos

C al conjunto de todas las ciudades

1 a la ciudad inicial

El resto de las ciudades están numeradas de 2 a n

Complejidad total:

Cantidad de subproblemas * calculo del costo subproblemas (lineal)

$O(n^2 2^n)$

```
Desde i=2 a n      //ciudad 1 es la inicial
    OPT[i][∅] = W[i][1]

Desde k=1 a n-2
    Para todo subset S de C-{1} de tamaño k

        Para cada elemento i de S

            OPT[‘i’,S-{i}] = +∞

            Por cada elemento j de S-{i}
                r=OPT[j,S-{i,j}] + w[j][i]

                Si (r<OPT[‘i’,S-{i}])
                    OPT[‘i’,S-{i}]=r

CaminoMinimo = +∞
Desde j=2 a n
    ciclo = OPT[i,S-{1,i}] + w[1][i]
    Si (CaminoMinimo>ciclo)
            CaminoMinimo = ciclo
Retornar caminoMinimo
```

Conclusión

El problema del viajante de comercio

Utilizando fuerza bruta O(n!)

Utilizando programación dinámica $O(n^2 2^n)$

n	Fuerza bruta	Prog dínamica
1	1	2
2	2	16
3	6	72
4	24	256
5	120	800
10	3628800	102400
15	1307674368000	7372800
20	2,43290200817664E+018	419430400
50	3,04140932017134E+064	2,81474976710656E+018
100	9,33262154439442E+157	1,26765060022823E+034

No es polinómico

No se conoce una solución polinómica.

(si alguien la encuentra… será el hombre mas famoso (o infame) del mundo.

Presentación realizada en Abril de 2020