UNIVERSITAT DE BARCELONA

Programació Dinàmica

Algorísmica Avançada | Enginyeria Informàtica

Santi Seguí | 2021-2022

Programació Dinàmica

Problema de la motxilla

Tenim diversos objectes, amb el seu pes, i tenim la capacitat total de la motxilla.
Quins objectes hauríem de posar per tal portar el màxim valor ***v*** possible amb el mínim pes ***w***?

$$\text{maximize} \sum_{i=1}^{n} v_i x_i$$

$$\text{subject to} \sum_{i=1}^{n} w_i x_i \leq W \text{ and } x_i \in \{0, 1\}.$$

Donat un conjunt d'elements numerats de l'1 al n, cadascun amb un valor de pes w i un valor de v , juntament amb una capacitat de pes màxima W.

Programació Dinàmica

Problema de la motxilla - Solució amb Força bruta

- Donat que hi ha n elements, pot haver-hi 2^n possibles combinacions.
- Provar totes les combinacions ens assegura trobar la solució òptima
- La complexitat serà O(2^n). Problema intractable quan n és gran

Solució vista a Algorísmica

Programació Dinàmica

Problema de la motxilla

Greedy?

Solució ràpida però no assegura trobar la solució òptima

Programació Dinàmica

Problema de la motxilla

Solució amb DP?

Programació Dinàmica

Problema de la motxilla

- Com resoldre el problema amb programació dinàmica?

    1. *Definir funció de recursivitat*
    2. *Trobar sub-estructura òptima*
    3. *Afegir casos base*

Programació Dinàmica

Problema de la motxilla

- Definició del sub-problema:
  - Si els articles s'etiqueten $1,\ldots,n$, el sub-problema per trobar una solució òptima per a $S_k = \{item\ \ etiquetats\ \ 1,2,\ldots,k\}$
    - La qüestió és, podem definir la **solució final** (S_n) en termes dels **sub-problemes** (S_k)?

Programació Dinàmica

Problema de la motxilla

Objectiu: Màxim benefici donat un
Pes màxim = 20 Kg

Definim el subproblema:

Per S_4**:**

Pes Total = 14;
Benefici màxim: 20$

Definim el subproblema:

Per S_5**:**

Pes Total = 20;
Benefici màxim: 26$

Programació Dinàmica

Problema de la motxilla

Objectiu: Màxim benefici donat un
Pes màxim = 20 Kg

Definim el subproblema:

Per S_4**:**

Pes Total = 14;
Benefici màxim: 20$

Definim el subproblema:

Per S_5**:**

Pes Total = 20;
Benefici màxim: 26$

la Solució S_4 no és part de la solució S_5. Per tant el subproblema no està ben definint

Programació Dinàmica

- Definició del sub-problema:
  - calcular **V[k,w]**, per tal de trobar una solució òptima per a $S_k = \{item\ \ etiquetats\ \ 1,2,...,k\}$ dins una motxilla de amb capacitat **w**

Programació Dinàmica

- Definició del sub-problema:
  - calcular **V[k,w]**, per tal de trobar una solució òptima per a $S_k = \{item\ \ etiquetats\ \ 1,2,...,k\}$ dins una motxilla de amb capacitat **w**
  - **Qüestió?** Si assumim que coneixem $V[i,j]$, on $i = 0,1,...,k-1$, $j = 0,1,...,w$, com podem derivar $V[k,w]$?

Anem a trobar la funció recursiva

Programació Dinàmica

Problema de la motxilla - **Funció recursiva**

- El millor subconjunt S_k que té pes total $\leq w$, pot contenir l'element k.
  - **Primer cas**: $w_k > w$. L'element k **no pot ser part de la solució**, ja que si ho fos, el pes total seria $> w$.
  - **Segon cas**: $w_k \leq w$. L'element k **pot** ser part de la solució, i únicament l'agafem si forma part de la millor solució.

$$V[k,w] = \begin{cases} V[k-1,w] & \text{if } w_k > w \\ \max\{V[k-1,w], V[k-1,w-w_k]+b_k\} & \text{sino} \end{cases}$$

Programació Dinàmica

Problema de la motxilla

Solució per un màxim de 4 elements i una motxilla de màxim pes = 6Kg

Dibuixeu tots els càlculs que necessitaríeu calcular.

Article	Pes	Valor	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
A	1	1	0	1(A)														
B	1	2		??														
C	2	2																
D	4	10								?								
E	12	4																

Programació Dinàmica

Problema de la motxilla

Article	Pes	Valor	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
A	**1**	**1**	0	1(A)	1(A)	1(A)	1(A)	1(A)	1(A)	1(A)	1(A)	1(A)	1(A)	1(A)	1(A)	1(A)	1(A)	1(A)
B	**1**	**2**	0	2(B)	3(AB)	3(AB)	3(AB)	3(AB)	3(AB)	3(AB)	3(AB)	3(AB)	3(AB)	3(AB)	3(AB)	3(AB)	3(AB)	3(AB)
C	**2**	**2**	0	2(B)	3(AB)	4(BC)	5(ABC)	5(ABC)	5(ABC)	5(ABC)	5(ABC)	5(ABC)	5(ABC)	5(ABC)	5(ABC)	5(ABC)	5(ABC)	5(ABC)
D	**4**	**10**	0	2(B)	3(AB)	3(AB)	10(D)	12(BD)	13(ABD)	14(BCD)	15(ABCD)	15(ABCD)	15(ABCD)	15(ABCD)	15(ABCD)	15(ABCD)	15(ABCD)	15(ABCD)
E	**12**	**4**	0	2(B)	3(AB)	3(AB)	10(D)	12(BD)	13(ABD)	14(BCD)	15(ABCD)	15(ABCD)	15(ABCD)	15(ABCD)	15(ABCD)	15(ABCD)	15(ABCD)	15(ABCD)

Com trobar els elements de la motxilla?

- Tota l'informació es troba a la taula.
- **V[n,W]** és el benefici màxim dels elements que podem carregar dins la motxilla.
  - Siguin $i = n$ i $k = W$
    - Si $V[i,k] \neq V[i-1,k]$:
      - marcar l'element i com a element present dins la motxilla.
      - $i = i - 1$
      - $k = k - w_i$
    - Si no:
      - $i = i - 1$ # Assumim que l'element i no està dins la motxilla

		0	1	2	3	4	5	6	7
Empty	0	0	0	0	0	0	0	0	0
$v_1=2, w_1=3$	1	0	0	0	2	2	2	2	2
$v_2=2, w_2=1$	2	0	2	2	2	4	4	4	4
$v_3=4, w_3=3$	3	0	2	2	4	6	6	6	8
$v_4=5, w_4=4$	4	0	2	2	4	6	7	7	9
$v_5=3, w_5=2$	5	0	2	3	5	6	7	9	10

		0	1	2	3	4	5	6	7
Empty	0	0	0	0	0	0	0	0	0
$v_1=2, w_1=3$	1	0	0	0	2	2	2	2	2
$v_2=2, w_2=1$	2	0	2	2	2	4	4	4	4
$v_3=4, w_3=3$	3	0	2	2	4	6	6	6	8
$v_4=5, w_4=4$	4	0	2	2	4	6	7	7	9
$v_5=3, w_5=2$	5	0	2	3	5	6	7	9	10

Programació Dinàmica

Travelling Salesman Problem

Programació Dinàmica

Travelling Salesman Problem

- Problema: Trobar un recorregut de longitud mínima per a un viatger que hagi de visitar diverses ciutats i després tornar al punt d'inici, on la distancia existent entre cada parella de ciutats és coneguda.

- És a dir, donat un graf dirigit amb arestes amb cost positiu, es vol trobar un **circuit de longitud mínima** que **comenci i acabi** en el **mateix node passant** exactament un cop per a cada un dels **nodes restants.**

Programació Dinàmica

Travelling Salesman Problem

- Aquest problema consisteix en trobar el circuit hamiltonià

- Problema molt estudiant donat la gran complexitat computacional necessària i la gran quantitat d'aplicacions a la vida real.

Programació Dinàmica

Travelling Salesman Problem - Solució per força bruta

- Trobar la solució mitjançant una cerca exhaustiva implica mirar totes les possibles permutacions dels nodes ordenats, el que implica una complexitat de complexitat **O(n!)**.

	A	B	C	D
A	0	4	1	9
B	3	0	6	11
C	4	1	0	2
D	6	5	-4	0

Tour	Cost		
A B C D	18	C A B D	15
A B D C	15	C A D B	24
A C B D	19	C B A D	9
A C D B	11	C B D A	19
A D B C	24	C D A B	18
A D C B	9	C D B A	11
B A C D	11	D A B C	18
B A D C	9	D A C B	19
B C A D	24	D B A C	11
B C D A	18	D B C A	24
B D A C	19	D C A B	15
B D C A	15	D C B A	9

Programació Dinàmica

Travelling Salesman Problem

- La solució mitjançant programació dinàmica millora significament la complexitat en el temps, passem d'una complexitat $O(n!)$ a $O(n^2 2^n)$

	n=2	**n=5**	**n=10**	**n=20**
Força Bruta	2	120	36.288.000	$1.4e^{18}$
DP	16	800	102.400	419.430.400

- La idea principal consisteix en calcular la solució òptima per a tots els subcamins de longitud N utilitzant informació que ja coneixem de les solucions parcials òptimes amb longitud N-1

-

Programació Dinàmica

Travelling Salesman Problem - Solució amb força bruta

Programació Dinàmica

Travelling Salesman Problem - Solució amb força bruta

Programació Dinàmica

Travelling Salesman Problem - Solució amb força bruta

* camins mínims

Programació Dinàmica

Travelling Salesman Problem - Solució amb força bruta

Programació Dinàmica

Travelling Salesman Problem

Acabem de veure una solució per força bruta utilitzant una estratègia Top-DOWN

Programació Dinàmica

Travelling Salesman Problem

- **Pensem una solució Bottom-UP**

Programació Dinàmica

Travelling Salesman Problem

- **Pensem una solució Bottom-UP**

- Aquestes son les tres funcions més simples. Totes elles acaben al node **w**

Programació Dinàmica

Travelling Salesman Problem

- **Pensem una solució Bottom-UP**

Programació Dinàmica

Travelling Salesman Problem

- **Pensem una solució Bottom-UP**

Programació Dinàmica

Travelling Salesman Problem

- **Pensem una solució Bottom-UP**

Tenim problemes superposats!!! → PODEM aplicar DP

Solució amb programació Dinàmica

- La idea principal consisteix en calcular la solució òptima per a tots els subcamins de **longitud N** utilitzant informació que ja coneixem de les solucions parcials òptimes amb longitud N-1

Programació Dinàmica

Travelling Salesman Problem

- $D(V_i, S)$ és la longitud del camí mínim sortint del vèrtex V_i i passant per cadascun dels vèrtexs del conjunt S i tornat al vèrtex V_i.

- La funció de recurrència es pot definir com:

$$g(i, S) = \{L_{0i}\} \quad \text{si } S = \{\}$$

$$g(i, S) = Min_{j \in S}\{L_{ji} + g(j, S - \{j\})\}$$

- $g(i, S)$ serà la longitud del camí mínim sortint del vèrtex i que passa per tots els vèrtex del conjunt S i torna al vèrtex i

Programació Dinàmica

Travelling Salesman Problem

- Sub-estructura òptima?

Programació Dinàmica

Travelling Salesman Problem

```
function algorithm TSP (G, n)
  for k := 2 to n do
    C({k}, k) := d_{1,k}
  end for

  for s := 2 to n-1 do
    for all S ⊆ {2, . . . , n}, |S| = s do
      for all k ∈ S do
        C(S, k) := min_{m≠k,m∈S} [C(S\{k}, m) + d_{m,k}]
      end for
    end for
  end for

  opt := min_{k≠1} [C({2, 3, . . . , n}, k) + d_{k,1}]
  return (opt)
end
```

Exercici

- Mirem les crides amb 5 nodes

Programació Dinàmica

Longest Increasing Subsequence (LIS)

- El problema de la seqüència creixent màxima, consisteix amb trobar una subseqüència d'una seqüència donada, tal que els elements de la subseqüència estiguin ordenats, de menor a major, i la subseqüència sigui el més gran possible. Per exemple:
  - Donada la seqüència: 0, 8, 4, 12, 2, 10, 6, 14, 1, 9, 5, 13, 3, 11, 7, 15
    - Una subseqüència creixent màxima pot ser: **0, 2, 6, 9, 11 , 15**
  - Aquesta subseqüència té longitud 6, i la seqüència d'entrada donada no té cap subseqüència creixent de tamany 7. La subseqüència anterior no és l'única de longitud 6, sinó que podem tindre diverses com per exemple:
- La solució d'aquest problema amb programació dinàmica ve donat per la següent funció recurrent:
  - $L(i) = 1 + max(L_0, L_1, \ldots, L_i - 1)$
  - amb el cas base:
  - $L(0) = 0$

Programació Dinàmica

Longest Palindromic Subsequence

- Donada una seqüencia de text, trobar la longitud the palindom més llarg.
- Exemple: **B**B**ABCB**C**AB**
  - Solució: longitud 7 : BABCBAB

Programació Dinàmica

Cent Saving

- Quan arribo a la caixa registradora, poso tots els meus n articles a la cinta transportadora i espero fins que tots els altres clients de la cua que tinc davant meu estiguin atesos.

- Mentre espero, m'adono que aquest supermercat recentment va començar a arrodonir el preu total d'una compra al múltiple més proper de 10 cèntims (amb 5 cèntims arrodonit a l'alça). Per exemple, 94 cèntims s'arrodoneixen a 90 cèntims, mentre que 95 s'arrodoneixen a 100.

- És possible dividir la meva compra en grups i pagar les peces per separat. Em pregunto on col·locar els d divisors per minimitzar el cost total de la meva compra. Donat que s'esgota el temps, no reoordenem el elements de la cinta.