10/28/22 76/100 (1)

BIEN

PROBLEMA A:

$k_1 .. k_n$: kioscos a visitar (en este orden)

$c_1 .. c_n$: #alfajores que quiere el kiosco.

$P_1 .. P_n$: Precio dispuesto a pagar el kiosco i por c_i alfajores. (P_i)

CADA KIOSCO i puede comprar c_i alfajores o Nada

CADA CAMIÓN TIENE UN TANQUE DE 90 LTS DE COMBUSTIBLE.

$d(k_i, k_{i+1}) = 10$ lts.

Al llegar a un kiosco, los camiones pueden elegir si:

k_i → cargar combustible (y no vender) (costo $0)

→ Vender alfajores: → Nada

→ c_i (a precio P_i)

El camión no debe quedarse sin combustible

MÁx Ganancia que puede obtener un camión que sale con A alfajores, el tanque lleno y los datos k_i, P_i, c_i?

(a) Se me ocurre plantear la sig. función recursiva

$f(i, a, t)$:= Máx Ganancia posible visitando los kioscos $k_i .. k_N$ y cumpliendo las restricciones.

La solución del problema sería $f(1, A, 90)$ y al plantearse como "solución subyo" vemos que los casos base son $i = N$ % $i = N+1$.

1, SI ES O NO FUNCIONA

definición

$f(i, a, t) :=$
- $-\infty$ -- si $t < 0 \,||\, a < 0$
- 0 -- si $i = N+1$
- $f(i+1, A, 90)$ -- si $A < C_i$ (para ahorrar llamados)
- MÁX{ -- cc
  - $f(i+1, A - C_i, T-10) + P_i$, --(vende)
  - [illegible] (aclarado en demo) [illegible]
  - $f(i+1, A, 90)$ -- (carga combustible y no vende)

  }

Hubiese sido más directo hacer $P(i)$ con i DEM | AO

$P(j)$ = MÁX GANANCIA de $k_{(n+1)-j} \cdots k_n$ cumpliendo las restricciones es $f(n+1-j, A, 90)$.

Dem correctitud

La veo por inducción en j:

* BASE: Si $i = N+1 \Rightarrow$ no hay kioscos para recorrer, luego la MÁXIMA ganancia es 0.

En particular esto vuelve a $P(0)$ válida porque $f(i+1, A, 90) = 0$ (si $A \geq 0$) $= \emptyset$ $(n+1)-0$

$P(0) =$ "Máx. GAN recorriendo $k_{n+1} \cdots k_n$ es $f(i, A, 90)$"

$= 0$ $\qquad = 0$

Luego este caso es válido.

* Paso Inductivo: qvq $P(m+1)$ suponiendo como HI. $P(k)\ \forall k.\ 0 \leq k \leq m$ (Inducción completa)

NO HACÍA FALTA HACERLA COMPLETA, BASTA CON QUE VALGA $P(m)$

veamos el caso:

- Estamos en el kiosco $(n+1) - m = i$ y conocemos las máx ganancias de c/ caso en los kioscos anteriores

⟶ ⟶ ⟶

Scanned with CamScanner

BIEN 10/28/22 (2)

* Si $A < C_i \Rightarrow$ sólo puedo cargar nafta.
$\Rightarrow P(m+1) = f(m+1, m-m+1, A, 90) \overset{\text{SEC F}}{=} F(m+1-m, A, 90) \overset{\text{HI}}{=} P(m) \Rightarrow$ VALE $P(m+1)$

* Desde un kiosco hay 3 posibilidades: (si $A > C_i$)

— Vender C_i alfajores a precio P_i

$$A' = A - C_i$$
$$MGN' = MGN + P_i$$
$$t' = t - 10$$

OBS: Paradójicamente, A', MGN', t' representan a un kiosco anterior al t A, MGN, t $\{A_1 \cdots A_N\}$.

— No vender nada:

$$A' = A$$
$$MGN' = MGN$$
$$t' = t - 10$$

→ igual vuelta

— No vender nada y cargar nafta.

$$A' = A$$
$$MGN' = MGN$$
$$t' = 90$$

→ tanque lleno (vuelvo a este caso si $A < C_i$)

Acá podemos hacer otra observación.

SIN PERDIDA DE GENERALIDAD: cuando no vendemos siempre es más fructífero cargar combustible ya que esto es de costo 0 e IMPIDE que no podamos vender a causa del combustible

Luego, de estos 2 casos restantes me interesa obtener el máximo:

Por HI: $f(i+1, A, t) \equiv$ "MGN en $K_{i+1} \cdots K_n$"

Luego

$$f(i, A, t) \overset{i \le n+1}{\underset{t \ge 0}{\underset{A \ge 0}{=}}} \max \{ f(i+1, A - C_i, t-10) + P_i \text{ -- vender}, \; f(i+1, A, 90) \} \text{ -- cargo}$$

Scanned with CamScanner

OJO, COMO ESTÁ ESCRITO "f", NO SE VERIFICA QUE f(i+1, A-ci, t-10) SEA MGN, SÓLO PODÉS ASEGURAR QUE f(i+1, A, 90)=MG

+ Cuando vendo MGN' = MGN + Pi ∧ A' = A - ci ∧ t = t

- Cuando cayo MGN' = MGN ∧ A' = A ∧ t = 90

Luego los casos están representados en f.
y f es válida/correcta.

b) Hay superposición de problemas cuando

$$O(\#\text{EstadosPosibles}) \ll \Omega(\#\text{LlamadosRecursivos})$$

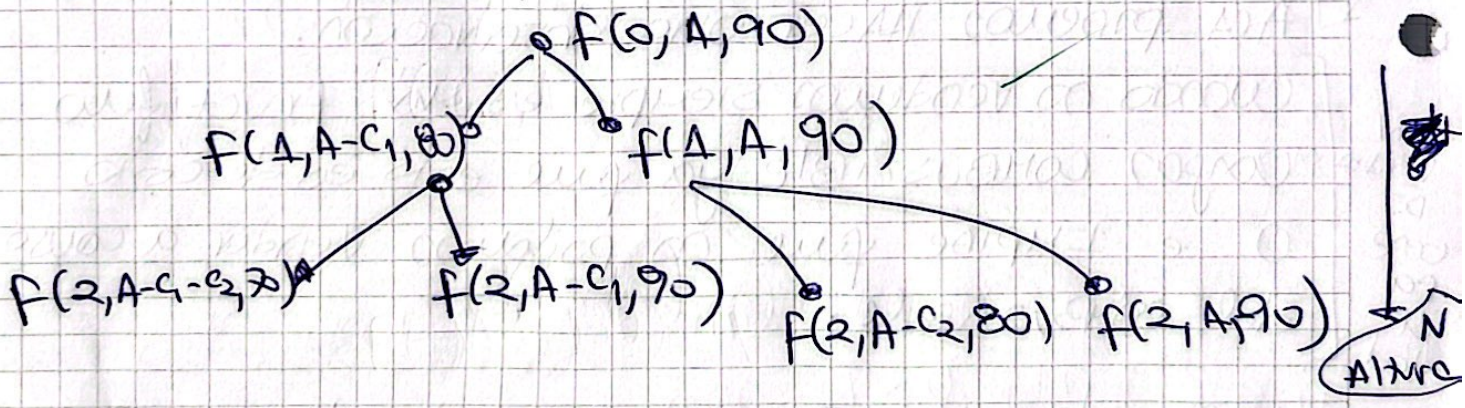
"Mis" estados posibles son:

$i \in \{1..N\}$
$a \in \{1..A\}$
$t \in \{1..90\}$

→ Sólo nos interesan estos dos porque 90 es una constante (para el análisis asintótico)

O sea $\#\text{EstadosPosibles} \in O(NA)$

La #LlamadosRecursivos la podemos ver como:



De base sabemos la cota superior: $O(2^N)$
pero necesitamos la inferior

Scanned with CamScanner

10/28/22 ③

BIEN

$\Omega(??)$. Sabemos que $a \geq 0$ sino cae en clase.

- Si $A < C_i \ \forall i \in \{1..N\}$. siempre hay un item factible $\Rightarrow$ no hay superposición.
- Si $A = 0 \rightarrow$ fin. Luego la min # iteraciones hasta que $A = 0$ es $A / \text{MAX}(C_i)$.
- Además, si $t = 0$ fin. Luego la min # iteraciones hasta que $t = 0$ es $t_0/10 = 10$

$90 \rightarrow 80 \rightarrow 70 \rightarrow \dots \rightarrow 10 \rightarrow 0$

Entonces hay

$$\text{Mín}\left\{ \frac{A}{\text{MAX}(C_i)}, N, \frac{N}{10} \right\} = \#\text{mín pasos hasta el fin.}$$

(log máx) (con suerte)

Luego,

$$\#\text{nodos} \in \Omega\left(2^{\text{MÍN}\left\{ \frac{A}{\text{MAX}(C_i)}, N/10 \right\}}\right)$$

Esto es $\gg$ a $O(nA)$ cuando $n > 10$.

y/o $A \gg \max_{0 < i < N+1}(C_i)$.

→

→

→

→

algoritmo PD → →

Scanned with CamScanner

©

Algoritmo PD

```
Memo ← new Array<int>[n][A][10](-1);
Solve(0, A, 90).

Def Solve(i, a, t) ≡        int res;
  if t<0 || a<0:
    return -∞
  if Memo[i][a][t] ≠ -1:
    return Memo[i][a][t]
  if i = N+1:
    res ← 0
  else if A < C[i]:
    res ← f(i+1, A, 90)
  else:
    res ← Max(Pi + f(i+1, A-Ci, t-10),
              f(i+1, A, 90))
  Memo[i][a][t] ← res
  return res.
```

ESTOS IFS TIENEN QUE IR AL REVÉS EN ORDEN, PORQUE EL PRIMERO SE PUEDE IR DE RANGO CUANDO i == n

---

Esto es $O(nA)$ temporal y espacial porque cada subproblema lo calculo 1 vez y como mucho hay $n \cdot A$ de subproblemas.
Además cada subproblema se calcula en $O(1)$ sin contar los llamados recursivos

$$O(n \cdot A) \cdot O(1) \equiv O(nA)$$

Scanned with CamScanner