Asesoría 1: Algoritmos Voraces

July 5, 2022

Ejercicio 3. Quiero dirigir un carro de una ciudad a otra a lo largo de una carretera. El tanque de combustible del carro tiene capacidad suficiente para cubrir c kilómetros, El mapa de la carretera indica la localización de los puestos de combustible. Queremos encontrar un algoritmo que garantize el viaje con el menor número de abastecimientos.

Mas formalmente, usted recibe un arreglo A de n números reales, cada uno de los $n-1$ primeros números indica los puntos de posibilidad de recarga y el último elemento indica el lugar de destino. Debe encontrar un subconjunto de los elementos de A con menor tamaño, que cumpla que nunca se le va a agotar la gasolina, es decir, $A[i+1] \leq A[i] + \texttt{c}$ para todo $i < n$.

Diseñe un algoritmo voraz. Analize su algoritmo, justificando que es correcto usando las propiedades de elección voraz y subestructura óptima.

Puntos clave:

- Conjunto de posibles puntos de recarga: A $= \{a_1, a_2, \ldots, a_n\}$.

  Debemos encontrar un conjunto $B \subseteq A$ de tamaño mínimo, tal que nunca se nos agotará la gasolina.

- Asumo que los puntos están ordenados de manera creciente. Asumo que dado un punto $x \in A$, $\lfloor x \rfloor$ me da el primer punto $a_i \in A$ tal que $a_i \leq x$. (Ilustrar). Además, asumo que dados dos puntos cualesquiera $a_i \in A$ y $a_{i+1} \in A$, $a_{i+1} - a_i \leq c$.

- Para este problema, asumiremos que empezamos con el tanque lleno, es decir, que el conjunto A tiene a un punto a_0 que significa que tenemos un punto de recarga justo donde empezamos. $A = \{a_0, a_1, a_2, \ldots, a_n\}$.

- Nuestro tanque tiene capacidad para c kilómetros.

1. Elección Voraz: Sea a_0 el punto de menor valor en A. Elijo $\lfloor a_0 + c \rfloor$.

2. Algoritmo Voraz:

Require: Un conjunto de puntos posibles de recarga $A = \{a_0, a_1, a_2, \ldots, a_n\}$. Notar que agregamos a_0.

Ensure: Un conjunto de puntos de recarga $B \subseteq A$ de tamaño mínimo de tal manera que nunca se nos acabará la gasolina.

Voraz(A)**:**

1: **if** $A == \emptyset$ **then**
2: **return** $\emptyset$
3: **end if**
4:
5: Sea a_0 el primer elemento de A
6: Sea $a_* = \lfloor a_0 + c \rfloor$
7: $A' = A - \{a_i : a_i < a_*\}$
8: **return** $\{a_*\} \cup$ **Voraz**(A')

3. Prueba de la elección voraz:

- Si $j - 1 = 0$, entonces $a_{j-1} = 0$.

Sea X una solución óptima.. Si $\lfloor a_0 + c \rfloor \in X$, entonces no tenemos nada que probar. Si $\lfloor a_0 + c \rfloor \notin X$, entonces existe un punto a_j en su lugar. Sea $X' = X - \{a_j\} \cup \{\lfloor a_0 + c \rfloor\}$, mostraremos que X' es también una solución óptima. Para esto, probaremos que todo punto $a_i \in A$ también está cubierto por X'. Esto significa que X', cubre las mismas estaciones de recarga que X, es decir, recarga el mismo número de veces o menos. Si $a_i \notin [a_{j-1}, a_j]$, entonces a_i está cubierto por $X' = X' - \{a_j\}$. Si $a_i \in [a_{j-1}, a_j]$ entonces:

$$a_0 \leq a_{j-1} \leq a_i \leq a_j \leq \lfloor a_0 + c \rfloor$$

Por lo tanto, a_i está cubierto por $[a_0, \lfloor a_0 + c \rfloor]$ y como $|X| = |X'|$, X' es también una solución óptima.

4. Prueba de la subestructura óptima

Sea X una solución óptima para A. Sea $X' = X - \{\lfloor a_0 + c \rfloor\}$, entonces X' es óptimo para $A' = A - \{a_0\}$. Supongamos por contradicción que X' no es óptimo, entonces existe un Y' tal que $|Y'| < |X'|$ y $|Y| = |Y' \cup \{\lfloor a_0 + c \rfloor\}| < |X' \cup \{\lfloor a_0 + c \rfloor\}| = |X|$, una contradicción.

Ejercicio 4. Dado un arreglo A de n números naturales, encontrar un arreglo B de tamaño n, que tenga a los elementos de A permutados y que minimize la suma $\sum_{i=1}^{n} iB[i]$.

Diseñe un algoritmo voraz. Analize su algoritmo, justificando que es correcto usando las propiedades de elección voraz y subestructura óptima.

Puntos clave:

- $A = \{a_1, a_2, \ldots, a_n\}$ y $B = A$, pero con los elementos de A permutados de alguna manera tal que $\sum_{i=1}^{n} iB[i]$ es mínimo.
- Definiremos $costo(B) = \sum_{i=1}^{n} iB[i]$.

1. Elección Voraz: Elegir el máximo elemento de A.

2. Algoritmo Voraz:

Require: Un arreglo A de n números naturales $\{a_1, a_2, \ldots, a_n\}$.
Ensure: Un arreglo B que tiene los mismos elementos del arreglo A, pero permutados de cierta manera que minimiza $\sum_{i=1}^{n} iB[i]$.

Voraz(A)**:**
1: **if** $A == \emptyset$ **then**
2: **return** $\emptyset$
3: **end if**
4: Sea a_* el máximo elemento de A
5: $A' = A - \{a_*\}$
6: **return** $\{a_*\} \cup$ **Voraz**(A')

3. Prueba de la elección voraz:

Sea X una solución óptima. Si $a_* \in X$ tal que $\forall_{a \in A - \{a_*\}} a_* \geq a$ (es decir, a_* es máximo en A), entonces no hay nada que probar. Suponga entonces que a_* no es el primero elemento de X. Entonces existe un número z que fue elegido antes que a_*, llamemos a esta solución óptima X. Entonces queremos probar que $X' = X.swap(z, a_*)$ es también una solución óptima. Para probar esto, mostraremos que $costo(X') \leq costo(X)$. Es decir, que mi elección voraz, hace que el costo se mantenga igual o que disminuya.

- Notar que $costo(X) = iz + \ldots + ja_*$, donde i < j.
- Notar que $costo(X') = ia_* + \ldots + jz$, donde i < j.

$$costo(X') - costo(X) = (ia_* + jz) - (iz + ja_*) \leq 0$$

$$i(a_* - z) + j(z - a_*) \leq 0$$

Notar que $(a_* - z) \geq 0$ y que $(z - a_*) \leq 0$

y como $i < j$:

$$i(a_* - z) + j(z - a_*) \leq 0$$

Por lo tanto, X' es también una solución óptima.

4. Prueba de la subestructura óptima:

Sea X una solución óptima para A, entonces $X' = X - \{a_*\}$ es una solución óptima para $A' = A - \{a_*\}$. Suponga por contradicción que X' no es óptimo, entonces existe un Y' tal que $costo(Y') < costo(X')$ y como $Y = Y' \cup \{a_*\}$, entonces:

$$costo(Y) = costo(Y' \cup \{a_*\}) < costo(X' \cup \{a_*\}) = costo(X)$$

lo cual es una contradicción.

Ejercicio 3 (8 ptos). Sea $\mathcal{I}$ un conjunto de n intervalos en la recta real. Decimos que un conjunto X de puntos *corta* $\mathcal{I}$ si cada intervalo en $\mathcal{I}$ contiene al menos un punto en X.

Queremos encontrar el conjunto más pequeño de puntos que corta $\mathcal{I}$. Diseñe un algoritmo voraz que encuentra un conjunto de ese tipo. Analize su algoritmo, justificando que es correcto usando las propiedades de elección voraz y subestructura óptima.

(a) Mencione cual es la elección voraz

(b) Escriba el pseudocódigo (no código) de su algoritmo voraz. Indique claramente qué recibe y qué devuelve su algoritmo. El algoritmo debe ser recursivo.

(c) Demuestre que su elección voraz es correcta. Enuncie a modo de lema y demuestre que su lema es correcto.

(d) Demuestre que su algoritmo tiene subestructura óptima. Enuncie a modo de lema y demuestre que su lema es correcto.

Deberá utilizar la siguiente notación:

Para los intervalos: $\mathcal{I} = \{[s_1, t_1], [s_2, t_2], \ldots, [s_n, t_n]\}$, donde $[s_i, t_i]$ es un intervalo con punta inicial s_i y punta final t_i. Para soluciones devueltas por el algoritmo o utilzadas en las demostraciones: variables X, Y, Y', X'.

Puntos clave:

- Ordenamos los pares del conjunto $\mathcal{I}$ por la punta final. Es decir, dado un conjunto $\mathcal{I} = \{[s_1, t_1], [s_2, t_2], \ldots, [s_n, t_n]\}$, entonces $t_1 \leq t_2 \leq \ldots \leq t_n$.

1. Elección Voraz: Elegir el menor t_i.

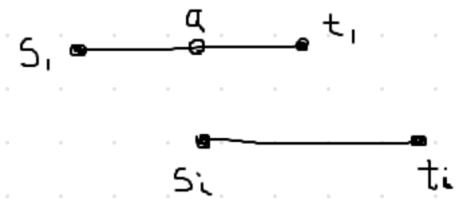
2. Algoritmo Voraz:

Require: Un conjunto $\mathcal{I} = \{[s_1, t_1], [s_2, t_2], \ldots, [s_n, t_n]\}$.
Ensure: Un conjunto X tal que $\forall [s_i, t_i] \in \mathcal{I}, \exists x \in X$ con $x \in [s_i, t_i]$.

Voraz($\mathcal{I}$)**:**
1: **if** $\mathcal{I} == \emptyset$ **then**
2: **return** $\emptyset$
3: **end if**
4: $\mathcal{I}' = \{[s_i, t_i] : s_i > t_1\}$
5: **return** $\{t_1\} \cup$ **Voraz**($\mathcal{I}'$)

3. Prueba de la elección voraz:

Sea X una solución óptima. Si $t_1 \in X$, entonces no tenemos nada que probar. Suponga entonces que $t_1 \notin X$. Como X es una solución, existe un punto $a \in X$ tal que $s_1 \leq a \leq t_1$. Sea $X' = X - \{a\} \cup \{t_1\}$. Probaremos que X' es también una solución óptima. Para ello, debemos demostrar que X' corta a $\mathcal{I}$ y que $|X| = |X'|$. Sea $[s_i, t_i] \in \mathcal{I}$. Si $a \notin [s_i, t_i]$, entonces existe un punto $b \in X$ tal que $s_i \leq b \leq t_i$. Dicho punto también está en X' y por lo tanto $[s_i, t_i]$ contiene algún punto de X'. Si $a \in [s_i, t_i]$, entonces:



$s_i \leq a \leq t_1 \leq t_i$ y por lo tanto, t_1 también está en $[s_i, t_i]$

. Luego, todo intervalo en $\mathcal{I}$ contiene al menos un punto en X' y X' corta a $\mathcal{I}$. Finalmente, como $|X'| = |X|$, X' es también una solución óptima.

4. Prueba de la subestructura óptima:

Sea X una solución óptima para $\mathcal{I}$, entonces $X' = X - \{t_1\}$ es una solución óptima para $\mathcal{I}' = \{[s_i, t_i] : s_i > t_1\}$. Suponga por contradicción que X' no es óptima en $\mathcal{I}'$. Entonces existe una solución Y' en $\mathcal{I}'$ tal que $|Y'| < |X'|$. Pero en ese caso $Y = Y' \cup \{t_1\}$ es una solución para $\mathcal{I}$ con tamaño $|Y'| + 1 < |X'| + 1 = |X|$, lo cual es una contradicción.