


87.  Considerem un model molt simplificat d'una xarxa de telefonia mòbil a zones rurals amb baixa densitat de població. Se'ns dona la ubicació de les n estacions base (antenes per a mòbil), especificades com a punts b_1, \dots, b_n al pla. També se'ns dona la ubicació dels n telèfons mòbils, especificats com a punts p_1, \dots, p_n al pla. Finalment, se'ns dona un paràmetre $\Delta > 0$ de distància de cobertura dels mòbils. Direm que el conjunt dels telèfons mòbils està *completament connectat* si és possible assignar a cada telèfon a una estació base de manera que:

- Cada telèfon s'assigna a una estació base diferent,
- Si el telèfon al punt p_i s'assigna a una estació base b_j , llavors la distància en línia recta entre p_i i b_j és $\leq \Delta$.

Suposem que el propietari del mòbil p_1 decideix fer un viatge en cotxe cap a l'est sense aturant-se, recorrent un total d' z unitats de distància. Com aquest telèfon mòbil es mou, s'ha d'anar actualitzant l'assignació del mòbil a diferents estacions base per tal de mantenir el mòbil connectat a la xarxa telefònica.

Doneu un algorisme polinòmic per decidir si és possible mantenir la connectivitat entre tot conjunt de mòbils en tot moment totalment connectats, durant el trajecte d'aquest telèfon. Podeu assumir que tots els altres telèfons romanen estacionaris durant aquest viatge. Si és possible mantenir la connectivitat, l'algorisme ha de produir una seqüència de l'assignació dels mòbils a les estacions base, que sigui suficient per a mantenir la connectivitat. Altrament, l'algorisme ha de tornar en quin punt (coordenada) del pla es perd la connectivitat. L'algorisme hauria de tenir una complexitat de $O(n^3)$.

Exemple: Suposem que tenim 2 mòbils a $p_1 = (0, 0)$ i $p_2 = (2, 1)$; i tenim 2 estacions-base a $b_1 = (1, 1)$ i a $b_2 = (3, 1)$ amb $\Delta = 2$. Suposem que p_1 es desplaça 4 unitats cap l'est fins al punt $(4, 0)$ aleshores podem mantenir la connectivitat entre els mòbils, al començament assignem $p_1 \rightarrow b_1$ i $p_2 \rightarrow b_2$ i quan p_1 arriba a $(2, 0)$ n'assignem $p_1 \rightarrow b_2$ i $p_2 \rightarrow b_1$.

Una solución.

Primero resolveremos el problema inicial antes de que el teléfono se empiece a mover.

Lo plantearemos como un problema de asignación con restricciones en una red de flujo en la que una unidad de flujo de s a t represente la asignación de un teléfono a una estación base.

La red \mathcal{N} tiene

- Nodos: s, t, P, B , $|P| = n$ representa teléfonos, $|B| = n$ representa las estaciones base.
- Aristas y capacidades:

$$\begin{array}{ll} \{(s, p) \mid p \in P\} & \text{capacidad 1} \\ \{(b, t) \mid b \in B\} & \text{capacidad 1} \\ \{(p, b) \mid p \in P, b \in B, d(p, b) \leq \Delta\} & \text{capacidad 1} \end{array}$$

Un camino de s a t tiene la forma $s \rightarrow p \rightarrow b \rightarrow t$, si transporta una unidad de flujo interpretaremos que el teléfono p se conecta con la estación base b .

Las conexiones (p, b) garantizan que el teléfono y la base están a la distancia correcta.

Si el problema tiene solución podremos asignar todos los teléfonos a bases dentro del radio permitido. Garantizando además que la asignación es 1 a 1.

Si traducimos cada asignación en una unidad de flujo en el camino asociado, obtenemos un flujo válido. Con valor de flujo n que es el máximo posible.

De la misma forma, si el flujo máximo en la red es n , reinterpretando el flujo como asignación, podemos asignar a cada teléfono una base, dentro del radio.

El flujo total en la arista (b, t) , de acuerdo con la ley de conservación de flujo, será 1, garantizando así que la asignación es 1 a 1.

Algorithmo:

```
Construir  $\mathcal{N}$   
 $F = \text{MaxFlow}(\mathcal{N})$   
if  $|F| < n$  then  
    return NO  
else  
    return  $\{(p, b) \mid F[p, b] = 1\}$ 
```

El coste de la llamada a MaxFlow: El número de vértices en la red es $N = 2n + 2$, el número de aristas es $M = 2n + n^2$.

- Con FF el coste es $O(n(N + M)) = O(n^3)$
- Con EK el coste es $O(M^2N) = O(n^5)$

Utilizaremos FF.

Construir \mathcal{N} tiene coste $O(N + M)$ y el último paso $O(n^2)$. El coste total del algoritmo es $O(n^3)$.

El teléfono p se mueve, los demás no

- Los únicos tiempos relevantes es cuando p pierde el contacto con la estación asignada. Como se mueve en una dirección fija solo puede pasar $n - 1$ veces.
- Cuando p pierde el contacto el resto de telefonos tiene asignada una base. Podemos utilizar esta asignación como flujo inicial en el algoritmo de FF, en vez de la asignación inicial de 0.

Algoritmo de reasignación:

Input: \mathcal{N} , flow F ,

Input telefono p , base b' , posición actual d

b' es la base asignada a p y $d(p, b) > \Delta$

Eliminar las aristas (p, b) de \mathcal{N}

Añadir las aristas (p, b) para las bases a distancia $\leq \Delta$ de d
con capacidad 1 y flujo 0

$F[s, p] = 0$ y $F[b', t] = 0$

$F = \text{MaxFlow}(\mathcal{N}, F)$

if $|F| < n$ **then**

return NO

else

return $\{(p, b) \mid F[p, b] = 1\}$

Com el valor del flujo F antes de la llamada a MaxFlow es $n-1$, solo podemos aumentar como máximo en una unidad el flujo. El coste es $O(N + M) = O(n^2)$.

El resto de modificaciones se puede hacer en tiempo $O(n)$.

Reasignar tiene coste $O(n^2)$ y lo haremos como mucho $O(n)$ veces.

El coste total del algoritmo es $O(n^3)$.