

## Problemes 4

- 4.1. 🚧 (SafeGym). ExtremeGym tiene que replanificar los circuitos de entrenamiento extremo incorporando las nuevas restricciones de seguridad sanitaria.

Para ello ha dividido los espacios en salas con capacidades limitadas. Algunas de estas salas se han conectado entre ellas a través de pasillos unidireccionales, debidamente aislados, en los que también se debe limitar el número de personas que los transitan.

Los técnicos de la empresa han diseñado un grafo en 5 niveles que define los posibles circuitos de entrenamiento. Un entrenamiento se inicia y finaliza en la zona de acceso al recinto (con capacidad ilimitada) que tiene acceso directo a todas las salas del primer nivel y desde todas las salas del último nivel. El entrenamiento finaliza tras realizar, en el tiempo estipulado, las rutinas asociadas a 5 salas, una en cada nivel. Los pasillos entre salas permiten acceder de una sala de entrenamiento en un nivel a otra (u otras) en el nivel siguiente.

ExtremeGym quiere un algoritmo eficiente para determinar el nivel de restricción que puede aplicar a la capacidad real de salas y pasillos que permita mantener un flujo mínimo de  $M$  personas realizando un entrenamiento en el gimnasio.

Diseñad un algoritmo eficiente para resolver el siguiente problema:

Dados el grafo de niveles y la capacidad de salas y pasillos, determinar si para algún valor entero no negativo  $k \geq 0$ , se puede limitar la capacidad de cada sala y cada pasillo en un factor  $2^k$  manteniendo un flujo de  $M$  entrenamientos. En el caso de que este valor exista proporcionar el valor de  $k$  más grande que permita el número de personas entrenando deseado.

**Una solución:** Lo resolveremos mediante un problema de flujo con restricciones. Tenemos restricciones en las salas y en los pasillos. La capacidad de un pasillo la trasladaremos a la capacidad de un arco. Para limitar la capacidad de las salas desdoblaremos los vértices correspondientes las salas en dos, de la forma habitual.

La entrada es  $G = (V, E)$  donde  $V = V_1 \cup V_2 \cup \dots \cup V_5$ . A partir de  $G$  consideramos la siguiente red de flujo. La red  $\mathcal{N}$  tiene

- Nodos:  $s, t, V, V' = \{v' \mid v \in V\}$  una copia de  $V$ ,
- Aristas y capacidades:

$\{(s, a) \mid a \in V_1\}$	capacidad $c(a)$
$\{(b, t) \mid b \in V_5\}$	capacidad $c(b)$
$\{(a, a') \mid a \in V\}$	capacidad $c(a)$
$\{(a', b) \mid a \in V, b \in V, (a, b) \in E\}$	capacidad $c(a, b)$

Un camino de  $s$  a  $t$  tiene la forma  $s \rightarrow a_1 \rightarrow a'_1 \rightarrow a_2 \rightarrow a'_2 \rightarrow \dots \rightarrow a'_5 \rightarrow t$ , si transporta una unidad de flujo interpretaremos que una persona realiza las rutinas de las salas involucradas. Las capacidades de las aristas aseguran que, en un flujo con valor máximo, en ningún momento se supera el aforo de una sala o la ocupación máxima de un pasillo.

Para resolver el problema primero obtendremos un flujo con valor máximo en la red, esto nos proporcionará el número máximo de personas que podrían entrar simultáneamente en el gimnasio sin violar las restricciones de seguridad.

Algorítmico:

```
Construir  $\mathcal{N}$ 
 $F = \text{MaxFlow}(\mathcal{N})$ 
return  $(|F|)$ 
```

El número de vértices en la red es  $N = 2n + 2$ , el número de aristas es  $M \leq 3n + m$ . Como el grafo tiene capacidades y no tenemos cotas sobre ellas utilizaremos EK. Así el coste es  $O(M^2N) = O(m^2n)$ .

Para calcular el valor de  $k$  que nos piden tengo en cuenta que la operación de escalado divide por el mismo factor todas las capacidades de la red, y por lo tanto el mismo factor de reducción aparecerá en el valor flujo máximo. Por ello, si  $MF$  es el valor devuelto por el algoritmo anterior tenemos que buscar el valor mayor de  $k$  tal que  $MF/2^k \geq M$ . Si  $MF < M$ , no se puede garantizar nunca el mínimo de ocupación. En caso contrario iremos dividiendo por 2  $MF$  y actualizando el valor de  $k$

```
Input  $MF(\geq M)$ 
 $k = 0$ 
while  $MF/2 \geq M$  do
     $++k$ 
     $MF = MF/2$ 
return  $k$ 
```

El número de iteraciones es  $O(\log MF)$  ya que siempre dividimos por dos. Así el coste total del algoritmo es  $O(m^2n + \log C)$  donde  $C = \sum_{a \in A_1} c(a)$  que es una cota superior a  $MF$ .

4.2. Les xarxes ad-hoc, composades per dispositius sense fils de baixa potència, s'han proposat per situacions com els desastres naturals en què els coordinadors dels treballs de rescat podrien controlar les condicions en zones de difícil accés. La idea és que una gran col·lecció d'aquests dispositius sense fils es podria llançar des d'un avió en una regió per a continuació reconfigurar-se com una xarxa operativa.

Estem parlant de: (a) dispositius relativament barats, el quals (b) es llancen des d'un avió a (c) un territori perillós; i per la combinació de (a), (b) i (c), es fa necessari fer front a la fallida d'un nombre raonable dels dispositius.

Ens agradaria que fos el cas que si un dels dispositius  $v$  detecta que està en perill de fallar, transmetés una representació del seu estat a un altre dispositiu a la xarxa. Cada dispositiu té un abast de transmissió limitat, pot comunicar-se amb altres dispositius que es troben com a màxim a  $d$  metres d'ell. Com que no volem que es transmeti el seu estat a un dispositiu inoperant, hem d'incloure una mica de redundància: Un dispositiu  $v$  ha de tenir un conjunt de  $k$  altres dispositius en un radi de  $d$  metres de distància. Anomenarem a aquest conjunt una *còpia de seguretat* per al dispositiu  $v$ .

Suposeu que després del llançament podem obtenir les coordenades  $p_i = (x_i, y_i)$  dels  $n$  dispositius operatius que formen la xarxa inicial.

Dissenyeu un algorisme per determinar si és possible triar una còpia de seguretat per a cada dispositiu, amb la propietat addicional que, per algun paràmetre donat  $b$ , cap dispositiu apareix en la còpia de seguretat de més de  $b$  altres dispositius. L'algorisme ha de proporcionar com a sortida també els conjunts de còpia de seguretat, sempre que es puguin trobar.

**Una solución:** Resolveremos el problema como un problema de flujo máximo. Para ello construimos una red  $\mathcal{N}$  forma

- Nodos  $V = \{s, t, u_1, \dots, u_n, v_1, \dots, v_n\}$
- Aristas y capacidades:
  - Para  $i \in [n]$ ,  $(s, u_i)$  con capacidad  $k$  y arco  $(v_i, t)$  con capacidad  $b$ .
  - Para  $i, j \in [n]$  con  $i \neq j$  y  $d(p_i, p_j) \leq d$ ,  $(u_i, u_j)$  con capacidad 1.

Supongamos que podemos seleccionar copias de seguridad  $C_i$  para todo  $i \in [n]$  verificando las restricciones. En este caso, para  $j \in C_i$  sabemos que  $d(p_i, p_j) \leq d$  y por tanto  $(u_i, v_j) \in E$ . Definimos la siguiente asignación de flujo: para  $i \in [n]$   $f(s, u_i) = k$  y para  $(u_i, v_j) \in E$ ,  $f(u_i, v_j) = 1$  si  $j \in C_i$ ,  $f(u_i, v_j) = 0$  si  $j \notin C_i$ ; para  $j \in [n]$ ,  $f(u_j, t) = \sum_{i \in [n], i \neq j} f(u_i, v_j)$ . Al cumplirse las restricciones, la asignación  $f$  es un flujo válido con valor  $kn$ .

Supongamos que en la red tenemos un flujo válido  $f$  con valor  $kn$ , definimos para  $i \in [n]$  el conjunto  $C_i = \{j \mid (u_i, v_j) \in E \text{ and } f(u_i, v_j) = 1\}$ . Como el valor del flujo es  $kn$ , cada vértice  $v_i$  recibe  $k$  unidades de flujo, por la ley de conservación del flujo, tenemos que  $|C_i| = k$ . Para la segunda restricción, observemos que  $|\{i \mid j \in C_i\}|$  coincide con el flujo de entrada en  $v_j$ , de nuevo por la ley de conservación de flujo esta cantidad tiene que coincidir con  $f(v_j, t)$ . Finalmente al ser un flujo válido tenemos  $|\{i \mid j \in C_i\}| = f(v_j, t) \leq c(v_j, t) = b$  y se cumple la segunda restricción.

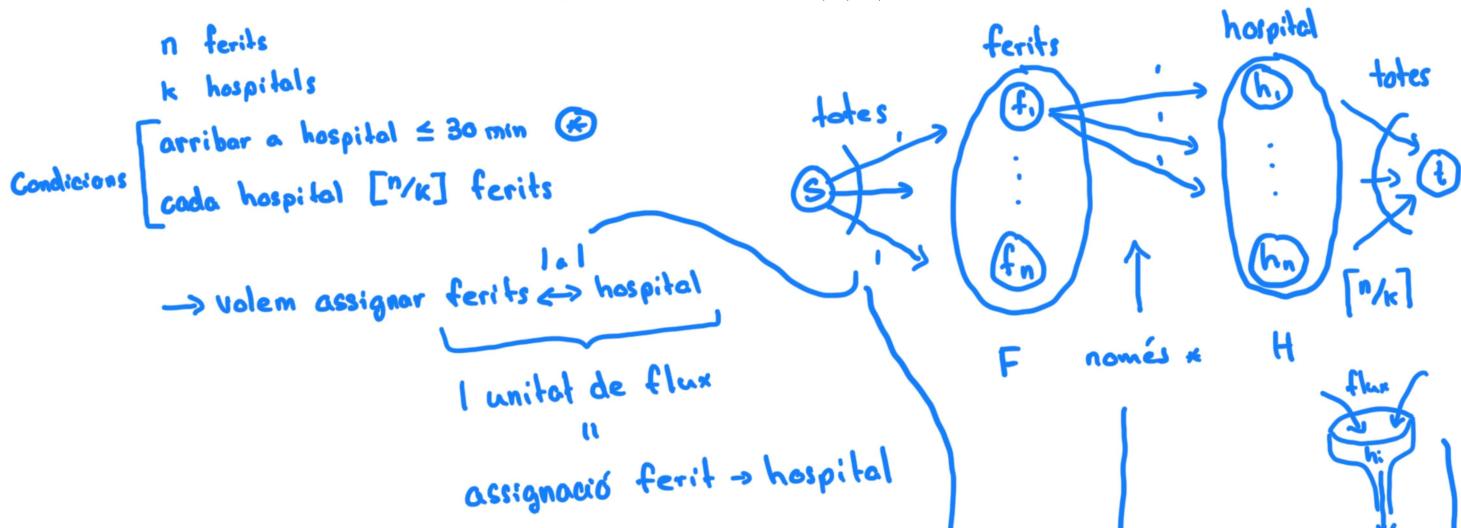
Por lo tanto tenemos que el problema planteado tiene solución si y solo si el valor del flujo máximo en la red construida es  $kn$ .

Por tanto para resolver el problema calcularemos un flujo con valor máximo utilizando un algoritmo que permita obtener soluciones enteras. Utilizando Ford-Fulkerson el número máximo de augmentations es  $kn$  y podemos resolver el problema en  $O(kn(|V| + |E|))$  es decir  $O(kn^3)$ .

Una vez obtenido este flujo  $f$ , para  $i \in [n]$  obtenemos el conjunto  $C_i = \{j \mid (u_i, v_j) \in E \text{ and } f(u_i, v_j) = 1\}$  con un recorrido de los arcos de  $G$  en  $O(n^2)$  time.

4.3. Considereu el següent escenari. Ha succeït una catàstrofe a una ciutat gran, els equips de rescat tenen identificats  $n$  ferits greus a diferents llocs de la ciutat, que necessiten ser traslladats urgentment a un hospital. Hi ha  $k$  hospitals disponibles. A causa de la gravetat de les ferides, és important que cada ferit arribi a un hospital abans de 30 minuts. Depenen a on el ferit està situat necessita anar a un hospital que estigui com a màxim a 30 minuts de distància per ambulància. Imagineu que vosaltres sou els responsables de la logística per traslladar els ferits als hospitals. Per a no col·lapsar urgències, voleu seleccionar els hospitals de manera que cada hospital rebi com a màxim  $\lceil n/k \rceil$  ferits.

Dissenyeu un algorisme polinòmic per què un cop rebeu la informació sobre el lloc on és cada ferit, pugueu determinar si és possible que tots els ferits siguin transportats a un hospital de manera que arriben abans de 30 minuts i que cap hospital rebi més de  $\lceil n/k \rceil$  ferits.



La xarxa  $N$  té:

- nodes:  $s, t, F, H$   
on  $|F|=n$ ,  $|H|=k$

- arcs i capacitat:

- $\{(s, f) \mid f \in F\}$  capacitat 1
- $\{(h, t) \mid h \in H\}$  capacitat  $\lceil n/k \rceil$
- $\{(f, h) \mid f \in F, h \in H, d(f, h) \leq 30\}$  capacitat 1

camí de flux:

$s \rightarrow f \rightarrow h \rightarrow t$

|||

$f$  va a l'hospital  $h$

SOLUCIÓ  $\Leftrightarrow$  TOTS els ferits s'han d'hospitalitzat

$\Updownarrow$

max-flow =  $n$

( $s$ : no no és solució)

Alg:  
 - construim  $N \rightarrow O(N+M)$   
 -  $F = \text{max-flow}(N) \leftarrow FF: O(n \downarrow \downarrow (N+M)) = O(n^2k)$   
 -  $EK: O(N+M^2) = O(n^2k^2(n+k))$   
 - if  $|F| < n$  then  
 return NO

else  
 return  $\{(f, h) \mid F[f, h] = 1\} \rightarrow O(nk)$

↳ per mirar quines aristes han quedat saturades per flux

la xarxa  $N$  té:  $(s, t)$

- $N = 2 + n + k$
- $N \leq n + k + nk$

4.4. El *problema de la evacuación* tiene la siguiente definición. Nos dan un grafo dirigido  $G = (V, E)$  que representa una red de carreteras. Una cierta colección de nodos  $X \subseteq V$  se designan como *nodos habitados*. Otra colección de nodos  $S$  son designados como *nodos seguros*. Suponemos que  $S$  y  $X$  son disjuntos. En el caso de una emergencia, queremos conocer rutas de evacuación desde los nodos habitados a los seguros. Un conjunto de *rutas de evacuación* se define como un conjunto de caminos en  $G$  tales que: (i) cada nodo en  $X$  aparece como el inicio de un camino, (ii) el último nodo en un camino aparece en  $S$ , (iii) los caminos no comparten aristas.

Cada ruta de evacuación proporciona una ruta a través de la cual los habitantes de un nodo poblado pueden escapar a un nodo seguro sin compartir ninguna parte del camino con habitantes de otros nodos.

Un conjunto de *rutas de evacuación mixtas* se define como un conjunto de caminos en el que (i) cada nodo en  $X$  aparece como el inicio de un camino, (ii) el último nodo en un camino aparece en  $S$ .

Este tipo de rutas proporciona rutas de escape en las que parte del camino es compartido entre habitantes de distintas ciudades.

Supongamos que además, debido a las restricciones de tráfico, cada arco  $e \in E$  tiene asignada una capacidad de tráfico  $c(e)$  que corresponde al número máximo de vehículos que pueden atravesarlo. Una petición de tráfico  $r(x)$ , para un nodo habitado  $x$ , representa el número de vehículos que se quieren evacuar desde  $x$ .

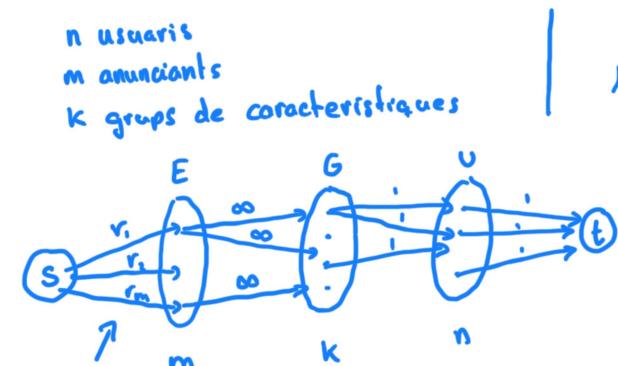
- (a) Dados  $G = (V, E, c)$  y  $S, X \subseteq V$ , muestra como decidir en tiempo polinómico si un conjunto de rutas de evacuación existe o no. En caso de que si que existan, el algoritmo debe proporcionar, para cada nodo poblado, la capacidad del arco de capacidad mínima de la ruta que se inicia en él.
- (b) Dados  $G = (V, E, c)$ ,  $S, X \subseteq V$ , y una petición de tráfico  $r(x)$ , para cada  $x \in X$ , muestra como decidir en tiempo polinómico si existe un conjunto de rutas mixtas de evacuación que permitan enviar los vehículos solicitados desde  $X$  a nodos seguros y que, además, para cada arco, se cumpla la condición de que el número de vehículos que lo atraviesan no supere la capacidad de tráfico del arco. En caso de que sea posible la evacuación con rutas mixtas, el algoritmo tiene que devolver además, para cada nodo  $x \in X$ , un plan de evacuación indicando: para cada vehículo saliendo de  $x$ , un camino con origen en  $x$  y final en un nodo seguro. El total de tráfico asignado entre todos los caminos del plan de escape de todos los nodos  $x \in X$  tiene que cubrir la totalidad de las peticiones de tráfico y cumplir las restricciones de capacidad de los arcos.

4.5. Una gran part del potencial d'empreses com Yahoo!, Google, Amazon, etc. es basa en el fet que milions de persones visiten cada dia les seves pàgines, (en angles aquest fet s'anomena "eyeballs"). Un cop una persona visita una pagina web d' aquestes companyes, de manera subliminal (o no tant) se intenta convèncer per a que deixen informació personal, la qual cosa permet que en el futur, si la mateixa persona torna a visitar la mateixa pagina web, rebi informació i anuncis extremadament personalitzats i adreçats a l'usuari. Per exemple, si l'usuari ha transmès informació a Yahoo!, que té 20 anys i estudia a la UPC, a Barcelona, rebrà tota mena d'informació sobre lloguers d'habitacions a Barcelona, acadèmies etc. D'altra banda, si l'usuari és un alt executiu rebrà anuncis sobre cotxes de luxe, creuers per illes exòtiques o vacances a la lluna. Una tasca algorísmica interessant és personalitzar els anuncis seleccionant els més adients a cada persona.

Suposem que els administradors d'una web popular han identificat  $k$  grups de característiques diferents (perfils)  $G_1, G_2, \dots, G_k$ , que no són necessàriament disjunts, per exemple  $G_i$  pot ser viure a l'Hospital,  $G_j$  ser dona i  $G_k$  que estudia a la UPC. La companyia propietària de la web té contractes amb  $m$  empreses anunciadores per mostrar un nombre determinat de còpies dels seus anuncis als usuaris de les seves pàgines web. El contracte que la companyia  $i$ -èsima fa amb Yahoo! és del tipus:

- Per a un subconjunt  $X_i \subseteq \{G_1, \dots, G_k\}$  dels grups demogràfics,  $i$  vol que els seus anuncis apareguin únicament a grups que són a  $X_i$ .
- Donat un enter  $r_i$  l'anunciант vol que els seus anuncis es mostren al menys a  $r_i$  usuaris.

Suposem que en un moment donat, hi ha  $n$  usuaris visitant la web. Com que tenim la informació de cadascun d'aquests usuaris, és fàcil conèixer el perfil de l'usuari  $j$  (per  $j = 1, 2, \dots, n$ ), es dir a quin subconjunt  $U_j \subseteq \{G_1, \dots, G_k\}$ , pertanyé Voldríem dissenyar un mecanisme tal que cada usuari vegi un únic anunci (d'uns pocs segons) de manera que per se satisfan restriccions imposades per els  $m$  anunciants? És dir, per a cada  $i = 1, 2, \dots, m$  com a mínim  $r_i$  usuaris, on cadascun pertanyé a almenys a un grup a  $X_i$ , veu un anunci proporcionat per l'anunciант  $i$ . Donar un algoritme eficient (polinòmic) per decidir si això és possible, i si és així, per a triar un anunci que aparegui per a cada usuari.



**Solució  $\Leftrightarrow$  TOTS ANUNCANTS ANUNCIEN MEN.  $r_i$ :**

$$\max\text{-flow} = \sum_{i=0}^m r_i \leq n$$

$$N = 2 + m + k + n$$

$$H \leq m + mk + kn + n$$

$$\begin{aligned} \textcircled{1} \text{ FF } O(C(N+N)) &= O(n(m+k+n)) \\ \textcircled{2} \text{ EK } O(N^2+N) &= O((m+mk+kn+n)^2 \cdot (m+k+n)) \end{aligned}$$

**Alg.:**

- Construir  $N' \Rightarrow O(N+N')$
- $F = \max\text{-flow}(N')$
- If  $|F| = \sum_{i=0}^m r_i$

return no  
else  
return f

entreure camins  
de flux  $s \rightarrow t$   
 $\Downarrow$   
assignació

La xarxa  $N'$  té:

- Nodes  $s, t, E, G, U$   
on  $|E|=m, |G|=k, |U|=n$
- Arcs i capacitat:
  - $\{(s,e) | e \in E\}$  capacitat  $r_i$
  - $\{(e,g) | e \in E \wedge g \in G \wedge g \in X_i\}$  capacitat  $\infty$
  - $\{(g,u) | g \in G \wedge u \in U \wedge \delta(g,u)\}$   $\delta(g,u)$  l'usuari  $u$  pertany al grup  $g$  capacitat 1
  - $\{(u,t) | u \in U\}$  capacitat 1

**Camí de flux**  
 $s \rightarrow e \rightarrow g \rightarrow u \rightarrow t$

anunciант mostra un anunci a un usuari  $u$  del grup  $g$

4.6. El servicio de control alimentario de la Generalitat está intentando acreditar un procedimiento de medida de la calidad alimentaria de productos complejos. En el laboratorio disponen  $m$  cromatógrafos y tienen técnicas de análisis de  $n$  componentes. Para cada cromatógrafo  $j$  se conoce conjunto  $S_j$  de componentes que puede analizar.

Para acreditar el procedimiento necesitan garantizar que se han realizado como mínimo  $k$  análisis en cromatógrafos diferentes de cada una de las componentes. Dependiendo del nivel de independencia se requieren condiciones adicionales.

- Nivel A: Un cromatógrafo solo puede utilizarse para analizar un máximo de dos componentes.
- Nivel B: Los cromatógrafos del laboratorio son de tres marcas diferentes. En este nivel se requiere además que los cromatógrafos que analicen una componente no sean todos de la misma marca.

Diseñad algoritmos para resolver los siguientes problemas:

- Dados  $n, m, k$  y para cada cromatógrafo  $j$ ,  $1 \leq j \leq m$ , los conjuntos  $S_j \subseteq \{1, \dots, n\}$ , determinar si es posible realizar análisis de las  $n$  componentes con el nivel de acreditación A.
- Dados  $n, m, k$  y para cada cromatógrafo  $j$ ,  $1 \leq j \leq m$ , los conjuntos  $S_j \subseteq \{1, \dots, n\}$  y la marca  $m_j \in \{1, 2, 3\}$ , determinar si es posible realizar análisis de las componentes con el nivel de acreditación B.

Los dos algoritmos, siempre que sea posible, deben obtener además una asignación de cromatógrafos a componentes verificando las condiciones requeridas por el nivel de acreditación.

- 4.7. Dissenyeu un algoritme per a resoldre en temps polinòmic el següent problema. Donat un graf dirigit  $G = (V, E)$ , dos vèrtexs  $x, y \in V$  i un enter no negatiu  $p$ , determinar el mínim enter  $k$  tal que podem garantir que, en  $G$ , hi ha almenys  $p$  camins de  $x$  a  $y$  tals que una aresta de  $E$  no forma part de més de  $k$  tals camins.

- 4.8. SuperFast és una empresa de transport que està intentant decidir si li interessa participar en una oferta de treball de Amazon a Barcelona. Amazon voldria subcontractar el transport de productes de proximitat a SuperFast. El transport ha de garantir uns compromisos de puntualitat molt estrictes i SuperFast vol una estimació de la quantitat de nous vehicles que hauria d'incorporar a la seva flota i una estimació del cost corresponent al seu ús. Amazon li ha proporcionat els resultats de les seves simulacions de comportament dels clients i, en particular, de les necessitats de transport diàries per uns quants dies. Per un dia es disposa d'una llista de sol·licituds de transport. Cada sol·licitud de transport especifica les coordenades GPS de l'origen i del destí d'un enviament juntament amb l'hora de recollida i d'entrega. SuperFast disposa d'un programa que li proporciona el temps necessari de desplaçament d'un vehicle entre qualsevol parell de posicions de la ciutat coneixent el temps d'inici del trasllat.

En aquest primer estudi SuperFast assumeix el cas pitjor en el què els vehicles no poden portar més d'un enviament. A més, el vehicle ha de ser a la posició d'origen al temps estipulat i no pot deixar el punt de destí fins el temps estipulat de trasllat. Així, un vehicle que transporti un enviament pot encarregar-se d'un altre sempre que el temps de desplaçament entre el destí i el nou origen li permeti arribar l'hora estipulada. També assumeix que l'estimació del temps necessari de desplaçament és acurada.

Diseneu algorismes amb cost polinòmic que:

- Donats un nombre  $k$  i les sol·licituds de transport d'un dia, determini si es poden servir amb  $k$  vehicles.
- Donades les sol·licituds de transports d'un dia, determini el nombre mínim de vehicles que les poden servir ( $k_{\min}$ )
- Donades les sol·licituds de transports de un dia, proporcioni una planificació per a cadascun dels  $k_{\min}$  vehicles indicant, per a cada vehicle, la seqüència de sol·licituds de transport que ha de servir i el temps total de desplaçament del vehicle.

Podeu suposar que el càlcul del temps de desplaçament entre dos posicions té un temps constant i que entre l'hora de recollida i la d'entrega hi ha temps suficient per fer-hi el desplaçament amb puntualitat.

Ajut: Podeu pensar un vehicle com una unitat de flux i una sol·licitud de transport com un arc amb fites inferiors i superiors a la seva capacitat.

### Una solució

- Utilizando la ayuda identificaremos los vehículos con unidades de flujo y las solicitudes con arcos con capacidad limitada con cota superior e inferior.

De las restricciones del problema sabemos que un vehículo puede servir otro envío siempre que pueda desplazarse con tiempo suficiente desde el destino al nuevo origen, asumiendo que permanece en el destino hasta la hora estipulada. Esta restricción nos indica cual es la red a considerar.

Tendremos, para cada solicitud  $i$ , dos nodos  $o_i$  y  $d_i$  y un arco  $(o_i, d_i)$  con capacidad  $[1, 1]$ , reflejando el hecho de que todas las solicitudes tienen que ser servidas. Añadiremos un arco  $(d_i, o_j)$  cuando saliendo del destino de la solicitud  $i$  en el tiempo estipulado podemos llegar al origen de la solicitud  $j$  en el tiempo estipulado, reflejando la observación previa.

Añadiremos tres vértices adicionales  $s, x, t$  y los arcos  $(s, x)$  con capacidad  $k$ , para forzar que el flujo máximo sea  $\leq k$ , y, para cada solicitud  $i$ , los arcos  $(x, o_i)$  y  $(d_i, t)$  con capacidad 1.

Notemos que la existencia de un flujo que satisface las restricciones de capacidad nos garantiza la existencia de  $\leq k$  caminos que no comparten aristas de  $s$  a  $t$  que, además, cubren todos los arcos correspondientes a solicitudes. Esto es equivalente a poder servir las peticiones con  $k$  o menos vehículos.

Una vez obtenida la red de flujo, podemos determinar si hay un flujo que satisface las restricciones en tiempo polinómico utilizando el algoritmo visto en clase para decidir la existencia de flujos con demandas y cotas inferiores.

- (b) Siempre hay una solución con  $n$  vehículos, por ello  $1 \leq k_{\min} \leq n$  podemos implementar una búsqueda binaria utilizando el algoritmo del apartado (a) que nos permite decidir si las solicitudes se pueden servir con  $\leq k$ . En total tendremos que hacer  $O(\log n)$  ejecuciones del algoritmo del apartado (a).
- (c) Una vez encontrado el valor de  $k_{\min}$  utilizando el algoritmo del apartado (a) para este valor de  $k$  tendremos una flujo de  $s$  a  $t$  con valor del flujo  $k_{\min}$  que satisface la restricción de que todas las solicitudes están servidas. Como hemos visto en clase lo único que tenemos que hacer es aplicar el algoritmo de extracción de caminos disjuntos como vimos en clase. Esto nos da  $k_{\min}$  ejecuciones de BFS sobre el grafo formado por los arcos con flujo positivo.

- 4.9. En sociologia, sovint s'estudia un graf  $G = (V, E)$  en el qual els nodes representen les persones, i les arestes representen els que són amics entre si. Suposem que l'amistat és simètrica, per la qual cosa podem considerar  $G$  com un graf no dirigit.

Volem estudiar aquest graf  $G$ , per a trobar grups de personnes, molt unides en el sentit de que tots són amics de tots. Una manera de formalitzar aquesta idea seria la següent. Per a un subconjunt  $S \subseteq V$  sigui  $e(S)$  el nombre d'arestes dintre d' $S$ , és a dir, el nombre d'arestes que tenen tots dos extrems en  $S$ . Definim la *cohesió* d' $S$  com  $e(S)/|S|$ . en aquest tipus de grafs, un paràmetre natural podia ser el conjunt  $S \subseteq V$  amb cohesió màxima.

- Doneu un algorisme polinòmic que pren com a entrada  $G$  i un nombre  $\alpha$  racional, i determina si hi ha un conjunt  $S$  amb cohesió  $> \alpha$ .
- Doneu un algoritme polinòmic per trobar un conjunt  $S \subseteq V$  amb la màxima cohesió.

### Una solución

Antes de definir el algoritmo vamos a asociar el problema con un parámetro de una red de flujo. Construimos la siguiente red:

- Por cada  $\{u, v\} \in E$  añadimos dos arcos  $(u, v)$  y  $(v, u)$  con capacidad 1.
- Añadimos vertice  $s$  y para cada  $u \in V$  un arco  $(s, u)$  con capacidad  $d(u)$ .
- Añadimos vertice  $t$  y para cada  $u \in V$  un arco  $(u, t)$  con capacidad  $2\alpha$ .

Analizemos la capacidad de los  $s - t$  cortes  $(S, T)$ . Supongamos que  $S = \{s\} \cup A$  and  $T = \{t\} \cup B$ .

Notemos que  $c(\{s\}, V \cup \{t\}) = 2m$  y  $c(V \cup \{s\}, \{t\}) = 2\alpha n$ .

Si  $\alpha < m/n$ ,  $V$  tiene la cohesión requerida

En caso contrario  $2m < 2\alpha n$  y mincut  $\leq 2m$ .

Para un corte genérico la capacidad del corte es:

$$\begin{aligned} c(S, T) &= \sum_{u \in B} d(u) + \sum_{v \in A} 2\alpha + c(A, B) \\ &= 2\alpha|A| + 2e(B) + 2c(A, B) \\ &= 2\alpha|A| + 2m - 2e(A) \\ &= 2(m + \alpha|A| - e(A)) \end{aligned}$$

$c(S, T) > 2m$  iff  $\alpha|A| - e(A) < 0$  iff  $\alpha|A| < e(A)$  iff  $e(A)/|A| > \alpha$ .

Hay un conjunto con cohesion  $> \alpha$  iff min-cut  $< 2m$

- La red se puede construir en tiempo polinómico. Luego calculamos el valor de MaxFlow usando Edmonds Karp en tiempo  $O(nm^2)$ . Y hacemos la comparación final. Obteniendo un algoritmo con coste polinómico.
- Primero calcularemos el valor mas grande de  $\alpha$  ( $\alpha_{\max}$ ) para el que existe un conjunto con cohesión  $\alpha$ . Para ello combinaremos búsqueda binaria con el algoritmo del apartado anterior. Una vez obtenido el valor  $\alpha_{\max}$ , obtenemos un flujo  $f$  con valor máximo usando Edmonds Karp en tiempo  $O(nm^2)$  para  $\alpha_{\max}$ . A partir de  $f$  obtenemos el grafo residual y tomamo como  $A$  los vertices accesibles desde  $s$  en este grafo. De acuerdo con el teorema maxflow-mincut  $s \cup A$  define un corte con capacidad máxima y de acuerdo con el resultado anterior  $A$  tiene cohesión máxima. Como en el paso anterior el tiempo total es polinómico ya que la cohesión de un conjunto es un valor entre 0 y  $n$ .

- 4.10. Supposeu que una persona ens presenta una possible solució al problema del flux màxim a una xarxa de flux  $\mathcal{N}$ . Doneu un algorisme que en temps lineal ens determini si la solució proposada té valor màxim.

1- Construïm el graf residual.  $\Rightarrow O(|E|)$   
2- Comprovem si existeix un camí de  $s$  a  $t$ .  $\rightarrow$  BFS  $\Rightarrow O(|E|)$

- Si existeix  $\Rightarrow$  no té valor màxim
- Si no existeix  $\Rightarrow$  té valor màxim

- 4.11. Tenim un graf bipartit  $G = (V, E)$  i un matching  $M$ . Proporcioneu un algoritme amb cost  $O(|V| + |E|)$  per determinar si  $M$  té cardinalitat màxima.

Donat un graf bipartit  $G = \langle L \cup R, E \rangle$

- 1- Afegim  $s \in L$  i  $t \in R$  a  $V = L \cup R \cup \{s, t\}$
- 2- Afegim totes les arestes amb capacitat 1 entre  $s : L$  i totes les arestes amb capacitat 1 entre  $R : t$ .
- 3- Apliquem una iteració de EK  $O(|V| + |E|)$ , si trobem un camí d'augmentació no és de cardinalitat màxima.

- 4.12. Tenim un digraf  $G = (V, E)$  on cada  $e \in E$  té capacitat  $c(e) = 1$ , i amb una font  $s \in V$  i un sumider  $t \in V$ . També en donen un paràmetre  $k \in \mathbb{N}$ . Doneu un algorisme amb temps polinòmic per a resoldre el següent problema: Volem eliminar  $k$  arestes de  $G$  de manera que reduïm al màxim que puguem el flux  $s \rightarrow t$ . En altres paraules, volem trobar un  $F \subseteq E$  tal que  $|F| = k$  i el flux màxim  $s \rightarrow t$  en  $G' = (V, E - F)$  sigui el més petit possible.

1- Construir el graf residual

2- Executar un algorisme max-flow

3- BFS desde s

4- Totes les arestes que van desde un node visitat a un no visitat  
formen el min-cut

5- Eliminem  $\min(k, |\text{min-cut}|)$  del graf

$\downarrow$   
per si  $k > |\text{min-cut}|$

- 4.13. Sigui  $G = (V, E)$  un digraf, i suposem que cada  $v \in V$  té el mateix nombre d'arestes que entren que d'arestes que surten, o sigui  $|\{(u, v) : (u, v) \in E\}| = |\{(v, u) : (v, u) \in E\}|$ . Donat un enter  $k \geq 1$ , siguin  $x, y \in V$  tal que existeixen  $k$  camins disjunts (sense repetir arestes)  $x \rightarrow y$ . Es cert que també existeixen  $k$  camins disjunts  $y \rightarrow x$ ? Si la resposta és afirmativa, doneu una demostració, altrament doneu un contraexemple.

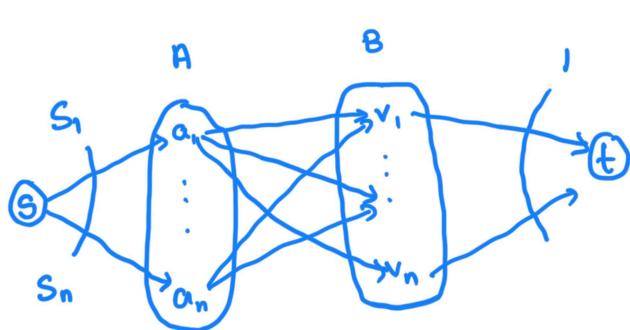
4.14. Un grup d'amics lloguen entre tots una embarcació d'esbarjo per sortir junts a navegar. Cada vegada que es fa un viatge cal que un dels usuaris (el mateix durant tot el viatge) s'encarregui de preparar el vaixell, n'assumeixi la conducció i l'atracada. També haurà de fer-se responsable de la neteja i reparació de desperfectes apareguts durant el viatge. A aquesta persona l'anomenem *responsible* del viatge. A ningú li ve de gust ser responsable i la nostra tasca és fer una assignació *justa* respecte de l'ús que fa cadascuna de les persones del grup.

Hi han dos elements que semblen raonables a l'hora de definir una assignació justa. Si una persona fa servir el vaixell moltes vegades, també hauria de ser responsable moltes vegades. D'altra banda, si una persona fa servir el vaixell en dies en què poques persones volen usar-lo, això també hauria d'incrementar el nombre de vegades que és responsable. Si no veus clar aquest segon criteri, pensa en el cas que 2 persones viatgen 20 vegades juntes i amb ningú més. És lògic que en siguin responsables 10 vegades cadascuna. Si per altra banda hi ha 20 persones que viatgen 20 vegades juntes, i amb ningú més, és natural que els toqui ser responsable un cop a cadascuna d'elles.

Suposem que una persona viatja  $r$  vegades. En el seu primer viatge hi ha  $k_1$  passatgers, en el segon  $k_2$ , i així fins  $k_r$ . Sembla just que a aquesta persona sigui responsable  $S = \sum_{j=1}^r \frac{1}{k_j}$  vegades. Com que aquest número pot no ser sencer, ens conformarem amb que sigui responsable com a molt  $\lceil S \rceil$  vegades. Si això passa per a tothom, direm que la assignació és justa.

Assumim que el grup està format per un conjunt  $A$  de  $m$  amics,  $A = \{1, \dots, m\}$ . L'entrada del problema correspon a la informació de  $n$  viatges. Per a cada viatge  $i$ , s'ens dona el conjunt  $V_i \subseteq A$  de viatgers.

- Demostreu que per a qualsevol entrada  $V_1, \dots, V_n$ , hi ha una assignació justa.
- Proporcioneu un algorisme de cost polinòmic en  $n$  i  $m$  que proporcioni una assignació de responsable per a cadascun dels  $n$  viatges que sigui justa.



$$S_i = \left[ \sum_{j=1}^{r_i} \frac{1}{k_{ij}} \right]$$

$$\begin{aligned} k_{ij} &= \# V_j \text{ si } a_i \in V_j \\ c(a_i, v_j) &= 1 \\ r_i &= \# \text{ de viatges de } a_i \end{aligned}$$

$(a_i, v_j) \in E$  si  $a_i$  ha fet

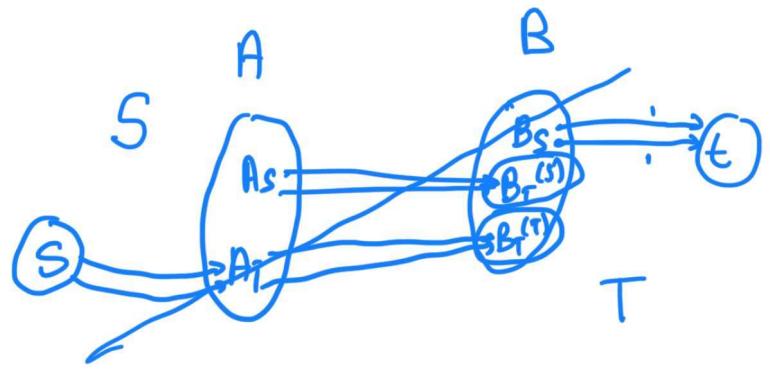
el viatge  $v_j$   $a_i \in V_j$

arcos  
(s,  $a_i$ )

$c(s, a_i) = S_i$

arcos  
( $v_j$ , t)

$c(v_j, t) = 1$



$$\forall S, T \quad n \leq c(S, T) \Rightarrow c^*(S^*, T^*) = v(f^*) \geq n$$

$$R = \sum_{i \in A_S} s_i = \sum_{i \in A_T} \left[ \sum_{j: a_i \in V_j} \frac{1}{|V_j|} \right] \geq \sum_{i \in A_T} \sum_{j: a_i \in V_j} \frac{1}{|V_j|} \geq \sum_{i \in A_T} \sum_{\substack{j: a_i \in V_j \\ V_j \subseteq B_T^{(s)}}} \frac{1}{|V_j|} =$$

$$= \sum_{V_j \in B_T^{(s)}} \frac{1}{|V_j|} \sum_{i \in A_T} 1 = \sum_{V_j \in B_T^{(s)}} \frac{1}{|V_j|} \sum_{a_i \in V_j} 1 = \# B_T^{(s)}$$

$$c(S, T) = \# B_S + R + (\geq \# B_T^{(s)}) = \# B_S + (\geq \# B_T^{(s)}) + (\geq \# B_T^{(t)}) \geq \# B = n$$

4.15. Suposeu que hi han  $n$  països que comercien entre ells. Per a cada país  $i$ , coneixem el valor del seu superàvit pressupostari  $s_i$ , (aquest nombre pot ser positiu o negatiu, un nombre negatiu indica un déficit). Per a cada parell de països  $i, j$  coneixem  $e_{i,j} \geq 0$ , el valor total de totes les exportacions d' $i$  cap a  $j$ , aquest nombre és sempre no negatiu. Diem que un subconjunt  $S$  dels països és *autosuficient* si la suma dels superàvits pressupostaris dels països a  $S$ , menys el valor total de tots els exportacions dels països a  $S$  cap a els països que no són a  $S$ , és més gran que zero.

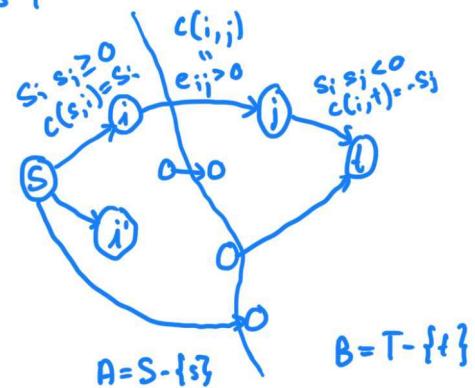
Doneu un algorisme polinòmic, que a partir d'aquestes dades per  $n$  països, decideix si existeix un subconjunt **no buit**, que és autosuficient.

$s_i =$  superàvit ( $\geq 0$ ) o déficit ( $< 0$ ) del país  $i$

$e_{ij} \geq 0$  = exportaciones de  $i$  a  $j$

$S \subseteq \{P_1, \dots, P_n\}$  autosuficiente

$$\left[ \sum_{i \in S} \left( s_i - \sum_{j \notin S} e_{ij} \right) \right] \geq 0$$



$$N = \sum_{s_i > 0} s_i$$

$$c(S, T) = N - \sum_{\substack{i \in A \\ s_i > 0}} s_i - \sum_{\substack{i \in A \\ s_i < 0}} s_i + \sum_{\substack{i \in A \\ j \in B}} e_{ij} = N - \left( \sum_{i \in A} s_i - \sum_{j \in B} e_{ij} \right)$$

$$c(S, T) = \sum_{i \in A, s_i < 0} (-s_i) + \sum_{i \in B, s_i > 0} s_i + \sum_{j \in B} e_{ij}$$

$$f^* < N \Rightarrow c^{**}(S^*, T^*) < N$$

$$s_i, s_j < 0 \rightarrow \exists S^* \quad c(j, t) = -s_j \quad \sum_{i \in S^* - \{s\}} s_i - \sum_{j \in T^* - \{t\}} e_{ij}$$

- 4.16. Suposeu que esteu organitzant un congrés on els investigadors presentin els articles que han escrit. Els investigadors que vulguin presentar un article envien un document als organitzadors de la conferència. Els organitzadors de la conferència tenen accés a un comitè de revisors que estan disposats a llegir-ne com a molt  $R$  articles cadascun. Tots els articles enviats han de ser revisat per, com a mínim,  $A$  revisors.

El congrés té declarats un conjunt de temes. Cada enviament té assignat un tema concret i cada revisor té declarada una especialització per a un conjunt de temes. Els articles sobre un tema determinat només es revisen per part dels revisors experts en aquell tema.

Els organitzadors del congrés han de decidir (sempre que es pugui) quins avaluadors revisaran cadascun dels articles o, equivalentment, quins articles seran revisats per cada revisor. Es demana:

Proporcioneu un algorisme eficient per a resoldre aquest problema d'assignació.

**Una solución.** Lo resolveremos como un problema de circulación en una red de flujo. La red tendrá un vértice por cada artículo  $\{v_1, \dots, v_n\}$  y un vértice por cada revisor  $\{r - 1, \dots, r_m\}$ . Añadimos las aristas  $(s, v_i)$  con cota inferior  $A$  y cota superior  $m$ , las aristas  $(r_j, t)$  con capacidad  $R$  y aristas  $(v_i, r_j)$  con capacidad 1, siempre que el revisor  $j$  sea experto en el tema del artículo  $v_i$ .

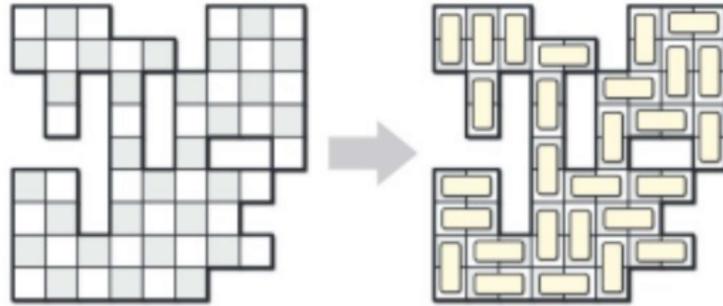
Una unidad de flujo que se transmita de  $s$  a  $t$ , pasará a través de un camino  $s \rightarrow v_i \rightarrow r_j \rightarrow t$  representando que el trabajo  $i$  se asigna al revisor  $j$ .

La capacidad 1 en las aristas  $(v_i, r_j)$  garantiza que no se asigna un artículo más de una vez a un revisor. La capacidad  $R$  en las aristas  $(r_j, t)$  evita que se asignen más de  $R$  artículos a un revisor. La cota inferior en las otras aristas garantiza que cada artículo tiene al menos  $A$  revisiones. La red admite una circulación si el problema planteado tiene solución.

Sea  $t(j)$  el número de temas en el que es experto el revisor  $j$  y  $T$  la suma de los valores de  $t(j)$ . La red tiene  $N = n + m + 2$  nodos Y  $M = n + T + m$  aristas.

El flujo máximo está acotado por  $mR$ , el coste será  $O(mR(n + m + T))$  si utilizamos la cota superior al flujo y  $O((n + m + T)^2(n + m))$  si utilizamos el tamaño de la red. Como  $R \leq n$  la mejor cota es  $O(mR(n + m + T))$ .

- 4.17. (Dòmino) Tenim un tauler  $n \times n$ , amb caselles de dos colors (blanc i negre) alternants per files i columnes. Esborrem un determinat nombre de caselles dels dos colors, de manera que al tauler resultant quedí el mateix nombre de caselles blanques i negres. Descriu i analitzeu un algorisme per a determinar eficientment si existeix una forma d'omplir el tauler amb fitxes de dòmino (que ocupen un àrea de  $2 \times 1$  caselles), de manera que totes les caselles que han quedat al tauler quedin cobertes i cap fitxa de dòmino surti del tauler.



A l'exemple de la figura, el tauler percolat de l'esquerra es pot omplir (completament i correctament) amb fitxes de dòmino; per tant, la resposta en aquest cas és positiva. Raoneu com, en cas de respostes afirmatives, es podria calcular la col·locació exacta de les fitxes de dòmino.

Ajut: Penseu que una fitxa de dòmino col·locada sobre el tauler sempre ocuparà una casella blanca i una casella negra que serà adjacent a la blanca. De totes les possibles caselles negres adjacents, l'algorisme haurà decidir quina.

#### Una solució:

Cobrir dues caselles amb una fitxa de dòmino és equivalent a ocupar una casella blanca i una negra que comparteixen algun costat. Per tant, cobrir el tauler percolat amb fitxes de dòmino és el mateix que aparellar totes les caselles blanques i negres de manera que totes les caselles queden aparellades i cap casella és part de més d'un parell.

Com a graf, podem representar totes les caselles negres com a nodes  $b_1, \dots, b_k$  i totes les caselles blanques com a nodes  $w_1, \dots, w_k$  (on  $k$  és el nombre de caselles blanques/negres que resten al tauler,  $k \leq \lceil \frac{n^2}{2} \rceil$ ).

Un arc  $(b_i, w_j)$  entre aquest dos grups de nodes correspon a la relació entre una casella negra  $b_i$  i les blanques que comparteixen costat (són adjacents) al tauler i, per tant, és possible aparellar-les entre elles sota una fitxa de dòmino. El resultat és un graf bipartit.<sup>1</sup> L'objectiu, donades aquestes restriccions, és trobar un *perfect matching* (o *aparellament*) a un graf bipartit, on cada casella formarà part d'un parell.

Per trobar el matching perfecte, ampliem aquest graf per transformar-lo en una xarxa de flux  $\mathcal{N}(V, E)$ . Com a nodes tindrem:

- un node font  $s$ ,
- els nodes  $b_1, \dots, b_k$  representant les caselles negres,
- els nodes  $w_1, \dots, w_k$  representant les caselles blanques,
- un node embornal  $t$ .

<sup>1</sup>Observeu que un node d'un determinat color mai tindrà cap arc cap a nodes del mateix color.

El nombre total de nodes de la xarxa és, doncs,  $2 + 2k \leq 2 + n^2 = O(n^2)$ .

Les arestes (dirigides) d'aquesta xarxa seran:

- $\{(s, b_i)\}_{1 \leq i \leq k}$
- $\{(w_j, t)\}_{1 \leq j \leq k}$
- $\{(b_i, w_j) \mid b_i \text{ és adjacent a } w_j\}_{1 \leq i \leq k}$

El nombre total d'arestes de la xarxa és  $|E| \leq 2k + 4k \leq 3n^2 = O(n^2)$ . Observem que, per a cada node  $b_i$  hi haurà, com a molt, quatre arestes  $(b_i, w_j)$ , corresponents a les quatre adjacències amb fitxes blanques que pot tenir com a màxim.

Considerem que totes les arestes tenen capacitat 1. Fixem aquesta capacitat perquè l'assignació d'una fitxa de dòmino ha de ser única per a cada casella (no es poden sobreposar fitxes) i una unitat de flux representarà precisament aquesta assignació.

Executem l'algorisme de Ford-Fulkerson per a determinar el flux màxim  $f^*$  de  $s$  a  $t$  en aquesta xarxa. Si hi ha un flux de mida  $k$ , aleshores hi ha un matching bipartit de mida  $k$ . El temps total de l'algorisme és, doncs,  $O(|E| \cdot f^*) = O(n^4)$  i, per tant, polinòmic.<sup>2</sup>

Per recuperar la col·locació exacta de les fitxes de dòmino només haurem de mirar quines arestes  $(b_i, w_j)$  han quedat amb flux ( $f[b_i, w_j] = 1$ ) després d'aplicar l'algorisme de Ford-Fulkerson (cost  $O(n^2)$ ).

---

<sup>2</sup>Atenció perquè, com ha de ser, estem expressant el cost de l'algorisme en funció de la mida de l'entrada. Recordeu que el tauler és de mida  $n \times n$ .

- 4.18. La compañía VideoFast quiere utilizar la red para transmitir vídeo a sus clientes. Para ello ha contratado una red de comunicación formada por  $n$  servidores que representamos con un grafo dirigido  $G = (V, E)$  con  $|V| = n$  vértices y  $|E| = m$  aristas.

Para cada  $e \in E$ ,  $b(e) \geq 0$  es el *ancho de banda* de la arista  $e$  contratado por VideoFast. El total de bits transmitido por un arco  $e$  no puede superar el ancho de banda contratado,  $b(e)$ . Por otra parte, VideoFast quiere establecer contratos con sus clientes, un subconjunto  $X \subset V$ . Para cada cliente  $x \in X$ , un contrato establece el tamaño total (en bits) de la transmisión  $t(x)$ . VideoFast quiere iniciar la transmisión desde un servidor  $s \in V - X$ .

Proporcionad un algoritmo con coste polinómico, para determinar si los anchos de banda contratados permiten efectuar o no la transmisión que se prevé establecer en los contratos con los clientes. En caso de que los contratos se puedan cumplir, el algoritmo debe proporcionar un nodo en  $V - X$  desde donde VideoFast pueda iniciar las transmisiones y cumplir con sus compromisos con los clientes.

### Una solución

Lo plantearemos como un problema de asignación con restricciones en una red de flujo en la que una unidad de flujo de  $s \in V - X$  a  $x \in X$  represente la transmisión de un bit de  $s$  a  $x$ . Para controlar la transmisión añadiremos un nodo  $t$  a la red y conectaremos  $X$  a  $t$ .

La red  $\mathcal{N}_v$ , para  $v \notin X$ , tiene

- Nodos:  $t$  y  $V$ , tomamos  $s = v$
- Aristas y capacidades:

$$\begin{array}{ll} e \in E & \text{capacidad } b(e) \\ \{(x, t) \mid x \in X\} & \text{capacidad } t(x) \end{array}$$

Un camino de  $s$  a  $t$  tiene la forma  $s \rightarrow \dots \rightarrow x \rightarrow t$ , si transporta una unidad de flujo interpretaremos que se envía un bit desde  $s$  a  $x \in X$ . La capacidad de las aristas que entran en  $t$  garantiza que se puede realizar la transmisión requerida si el flujo máximo en la red es  $\sum_{x \in X} t(x)$ .

Resolveremos MaxFlow, para cada  $v \in V - X$ , hasta encontrar un  $v$  tal que el valor del flujo máximo en  $\mathcal{N}_v$  es  $\sum_{x \in X} t(x)$ , y devolveremos este nodo, o comprobar que no es posible.

Las capacidades son enteras, pero los valores pueden ser grandes, por ello utilizarremos EK para calcular un flujo con valor máximo. Como tenemos que hacerlo para cada  $s \in V - X$ , el coste total del algoritmo es  $O(|V|^2|E|^2)$ .

### Una solución

Alternativamente se puede plantear el problema como un problema de circuación en vez de flujo.

La red  $\mathcal{N}_v$ , para  $v \notin X$ , tiene

- Nodos:  $V$
- Aristas y capacidades:

$$\begin{array}{ll} e \in E & \text{capacidad } b(e) \\ \{(x, t) \mid x \in X\} & \text{capacidad } t(x) \end{array}$$

- Demandas:  $d(v) = -\sum_{x \in X} t(x)$  y  $d(x) = t(x)$ ,  $x \in X$ .

### Una solución

Alternativamente se puede plantear el problema como un problema de circuación con cotas inferiores en vez de flujo.

La red  $\mathcal{N}_v$ , para  $v \notin X$ , tiene

- Nodos:  $t$  y  $V$ , tomamos  $s = v$
- Aristas y capacidades:

$$\begin{array}{ll} e \in E & \text{capacidad } b(e) \\ \{(x, t) \mid x \in X\} & \text{cotas } [t(x), t(x)] \end{array}$$

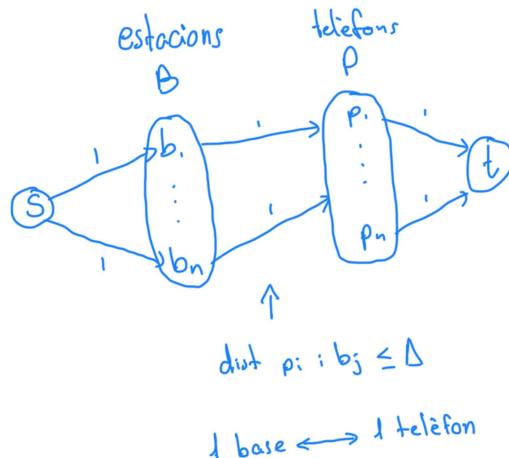
4.19. Considerem un model molt simplificat d'una xarxa de telefonia mòbil a zones rurals amb baixa densitat de població. Se'n dona la ubicació de les  $n$  estacions base (antenes per a mòbil), especificades com a punts  $b_1, \dots, b_n$  al pla. També se'n dóna la ubicació dels  $n$  telèfons mòbils, especificats com a punts  $p_1, \dots, p_n$  al pla. Finalment, se'n dóna 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 algoritme 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  $p_2 \rightarrow b_1$ .



La xarxa té:

- nodes:  $s, t, B, P$
- $|B| = |P| = n$
- arcs i capacitat
  - $\{(s, b) \mid b \in B\}$  capacitat 1
  - $\{(b, p) \mid b \in B \wedge p \in P\}$  capacitat 1
  - $\{(p, t) \mid p \in P\}$  capacitat 1

camí de flux:  
 $\boxed{s \rightarrow b \rightarrow p \rightarrow t} \rightarrow s \rightarrow t$

$$N = 2 + 2n = O(n)$$

$$M \leq 3n = O(n)$$

Alg:

$\boxed{O(n^3)}$

mentre  $p_i$  no punt final:  $\rightarrow O(n)$   
 construim  $W \rightarrow O(N+M) = O(n+n) = O(n)$   
 $F = \text{max-flow}(W) \rightarrow FF: O(C(N+M)) = O(n(n+n)) = O(n^2)$   
 si  $|F| < n$  llavors  $\rightarrow EK: O(N+M) = O(n+n) = O(n^2)$   
 return no  
 altresament  
 guardar f → extreure camins de flux s→t  
 return conjunt f

Solució  $\Leftrightarrow \forall$  camí de flux tot telèfon té assignat una base.

max-flow = n

4.20. MenjaBe produeix una gran varietat de menús de menjars diferents. Malauradament, només poden produir els seus menús en quantitats limitades, de manera que soLEN QUEDAR-SE SENSE ELS MÉS POPULARS, deixant clients insatisfets. Per minimitzar aquest problema, MenjaBe vol implementar un sofisticat sistema de distribució de dinars. Els clients haurien d'enviar un missatge de text amb les seves opcions de menús acceptables abans de l'hora de dinar. A continuació, fan una assignació de dinars als clients. MenjaBe té decidit compensar amb un val de 5 euros als clients als qui no pot assignar cap de les seves opcions. MenjaBe vol minimitzar la quantitat de vals que dóna.

- (a) Doneu un algorisme eficient per a assignar menús als clients en un dia. En general, en un dia determinat, MenjaBe sap que ha produït  $m$  tipus de menús i la quantitat de cadascun d'ells  $q_1, \dots, q_m$ . A més els  $n$  clients envien un text amb les seves preferències, el client  $i$  indica un conjunt  $A_i$  de menús acceptables. L'algorisme ha d'assignar a cada client una de les seves opcions o un val de 5 euros, de manera que es mimimitzi el nombre de vals.
- (b) Un dels directius de MenjaBe proposa minimitzar el diners gastats en vals oferint un descompte en el preu del menú als clients que optin per una opció de menú sorpresa. En aquest cas els clients poden rebre un qualsevol dels menús disponibles amb un descompte de 3 euros en el menú servit. En aquest cas els clients envian un mail indicant que volen un mnú sorpresa o la llista de les seves preferències. De nou, si no els poden servir un menú dintre de les seves preferències o, en els cas del menú sorpresa no els pot servir cap menu, rebran el val de 5 euros. Doneu un algorisme eficient per a determinar l'assignació menys costosa per a MenjaBe.

## 4.20

MENJABE produeix una gran varietat de menús diferents. Malauradament, només poden produir els seus menús en quantitats limitades, de manera que soLEN QUEDAR-SE sense els més populars, deixant clients insatisfets. Per minimitzar aquest problema, MENJABE vol implementar un sofisticat sistema de distribució de dinars. Els clients haurien d'enviar un missatge de text amb les seves opcions de menús acceptables abans de l'hora de dinar. A continuació, fan una assignació de dinars als clients. MENJABE té decidit compensar amb un val de 5 euros als clients als qui no pot assignar cap de les seves opcions. MENJABE vol minimitzar la quantitat de vals que dona.

- a) Doneu un algorisme eficient per a assignar menús als clients en un dia. En general, en un dia determinat, MENJABE sap que ha produït  $m$  tipus de menús i la quantitat de cadascun d'ells  $q_1, \dots, q_m$ . A més els  $n$  clients envien un text amb les seves preferències, el client  $i$  indica un conjunt  $A_i$  de menús acceptables. L'algorisme ha d'assignar a cada client una de les seves opcions o un val de 5 euros, de manera que es minimitzi el nombre de vals.

### Solució

Resolem el problema via càlcul de fluxos màxims. Definim la següent xarxa  $G = (V, E, c)$  amb capacitat  $c : E \rightarrow \mathbb{Z}^+ \cup \{0\}$ , on el conjunt de vèrtexos es defineix com

$$V = \{s, t, C_1, \dots, C_n, M_1, \dots, M_m\},$$

essent  $s$  el node font,  $t$  el node embornal,  $\{C_1, \dots, C_n\}$  els nodes que representen els  $n$  clients, i  $\{M_1, \dots, M_m\}$  els nodes que representen els  $m$  menús. El conjunt d'arestes  $E$  es defineix com

$$E = \{(s, C_i)\}_{1 \leq i \leq n} \bigcup \{(M_j, t)\}_{1 \leq j \leq m} \bigcup \{(C_i, M_j) \mid M_j \in A_i\}.$$

A les arestes de la font  $s$  als nodes  $C_i$  fixem capacitat 1. A les arestes dels nodes  $M_j$  a l'embornal  $t$  fixem capacitat  $q_j$  que correspon a la quantitat de menús de tipus  $M_j$  que MENJABE pot servir. A totes les arestes  $(C_i, M_j)$  fixem també capacitat 1.

Un cop construïda la xarxa, apliquem un algorisme de max-flow de manera que si  $f : E \rightarrow \mathbb{N}$  és el flux màxim calculat per l'algorisme, podem definir l'assignació òptima per a MENJABE

$$\text{assig} : \{C_1, \dots, C_n\} \rightarrow \{M_1, \dots, M_m\} \cup \{\text{val}\}$$

de la següent forma:

- $f(s, C_i) = 1 \implies \exists j : M_j \in A_i : f(C_i, M_j) = 1$

En aquest cas definim  $\text{assig}(C_i) = M_j$  amb cost per a MENJABE  $\text{cost}(C_i) = 0$ .

- $f(s, C_i) = 0 \implies \forall j : M_j \in A_i : f(C_i, M_j) = 0$

En aquest cas  $\text{assig}(C_i) = \text{val}$  amb cost per a MENJABE  $\text{cost}(C_i) = 5$ .

- $f(M_j, t) \leq q_j \implies$  mai se serveixen més menús dels que es disposa.

L'algorisme de max-flow maximitzà

$$|\{C_i \mid f(s, C_i) = 1\}|,$$

(el nombre de clients amb flow 1 des de la font) i per tant maximitza

$$|\{C_i \mid \text{cost}(C_i) = 0\}|,$$

(el nombre de clients que produeixen cost 0) i això vol dir que minimitza

$$|\{C_i \mid \text{cost}(C_i) = 5\}|$$

(el nombre de clients que produeixen cost 5 perquè se'ls proporciona un val) i minimitza per tant el nombre de vals que s'han de repartir. Així queda demostrada la correctesa.

Quin algorisme seria més adequat entre Ford-Fulkerson i Edmonds-Karp? Notem que el nombre de nodes és  $|V| = n + m + 2$ , i el nombre d'arestes és  $|E| = n + \sum_{i=1}^n |A_i| + m = O(nm)$ .

- Ford-Fulkerson:  $O(|E||f^*|) = O(n^2m)$ , perquè  $|f^*| \leq n$
- Edmons-Karp:  $O(|V||E|^2) = O(n^3m^2 + n^2m^3)$

Per tant, en aquest cas el més eficient seria el Ford-Fulkerson i caldria afegir el temps necessari per calcular l'assignació a partir del flux. En total,  $T(n, m) = O(n^2m)$ .

- b) *Un dels directius de MENJABE proposa minimitzar el diners gastats en vals oferint un descompte en el preu del menú als clients que optin per una opció de menú sorpresa. En aquest cas els clients poden rebre qualsevol dels menús disponibles amb un descompte de 3 euros en el menú servit. En aquest cas, els clients envien un email indicant que volen, o bé un menú sorpresa, o bé la llista de les seves preferències. De nou, si no els poden servir un menú dintre de les seves preferències o, en el cas dels menús sorpresa no se'ls pots servir cap menú, rebran el val de 5 euros. Doneu un algorisme eficient per a determinar l'assignació menys costosa per a MENJABE.*

## Solució

Construïm una xarxa com la de l'apartat a), considerant només els clients que tenen una llista de preferències (i no menú sorpresa). Calclem el flux màxim  $f_{\text{sorpresa}}^*$ , minimitzant així

$$|\{C_i \mid A_i \neq \{\text{sorpresa}\} \wedge \text{assig}(C_i) = \text{val}\}|.$$

Ara calclem:

- $Q = \sum_{i=1}^m q_j - |f_{\text{sorpresa}}^*|$  dels menús que no han estat assignats encara, i
- $S = |\{C_i \mid A_i = \{\text{sorpresa}\}\}|$  el nombre de clients que desitgen menú sorpresa.

Si  $Q \geq S$ , aleshores assignem un menú sorpresa a tots els clients que volien menú sorpresa. Cost mínim per  $\{C_i \mid A_i = \{\text{sorpresa}\}\}$ :  $3S$ .

Si  $Q < S$ , aleshores assignem tots els menús sorpresa possibles (en total  $Q$ ), i assignem un val a la resta (en total  $S - Q$ ), i aleshores el cost mínim per  $\{C_i \mid A_i = \{\text{sorpresa}\}\}$ :  $3S + 5(S - Q)$ .

No es poden barrejar els clients  $C_i$  amb  $A_i \subseteq \{M_1, \dots, M_m\}$  (clients amb conjunt de menús acceptables) amb els clients  $C_j$  amb  $A_j = \{\text{menú\_sorpresa}\}$ . Si a la xarxa tinguéssim al mateix temps aquests dos tipus de clients, el client “sorpresa” podria acceptar qualsevol menú. Indirectament, aleshores maximitzar el flux no implicaria minimitzar el cost.

L'eficiència de l'algorisme proposat segueix l'anàlisi de l'apartat a), afegint-hi l'assignació de menús sorpresa ( $O(n)$  addicional). En total,  $T(n, m) = O(n^2m)$ .

4.21. El Rector de la UPC de cara a fomentar la germanor entre el personal de la UPC ha decidit establir activitats mensuals per incentivar la interacció entre els diferents estaments de PDI i PAS. Les festes se celebraran normalment un cop al mes. Suposem que hi han  $k$  grups disjunts en el personal,  $C_1, \dots, C_k$ . El rectorat produeix una llista  $L$  d'activitats que tindran lloc en el curs acadèmic, i per a cada activitat  $a \in L$ , hi ha un nombre màxim  $M(a)$  i un nombre mínim  $m(a)$  de gent que pot assistir a  $a$ . A més, com que els grups tenen diferents nivells d'influència, el rectorat estima, per a cada  $j$ , el nombre mínim  $s(j)$  de persones del grup  $C_j$  que s'han de convidar a cada activitat. Per una altra banda és ben conegut que alguns membres de la UPC no poden participar sovint en activitats lúdiques. Tenint en compte aquest aspecte el rectorat ha determinat per cada membre  $i$  del personal de la UPC un valor  $t(i)$  indicant el nombre màxim d'activitats a les què  $i$  pot participar al llarg del curs. El problema consisteix en, en cas que sigui possible, decidir a qui convidar a cada activitat de manera que es compleixin les restriccions prèvies, o sigui:

- el nombre d'assistents a l'activitat  $a$  és  $\leq M(a)$  i  $\geq m(a)$ ,
- per a cada grup  $j$  hi ha d'haver com a mínim  $s(j)$  persones assistint a cada activitat.
- la persona  $i$  mai es convidada a més de  $t(i)$  activitats.

Proporcioneu un algorisme de cost polinòmic per aquest problema.

n persones  $P = \{p_1, \dots, p_n\}$

K grups disjunts de persones  $\{C_1, \dots, C_k\}$

L activitats  $L = \{a_1, \dots, a_m\} \quad \forall a \in L \quad m(a) \leq \text{persones}(a) \leq M(a)$

$s(j) \rightarrow$  min persones a convidar del grup  $C_j$  a cada activitat

$t(i) \rightarrow$  max activitats que pot participar una persona

ASSIGNACIÓ: persona  $\rightarrow$  activitat

Camí de flux:

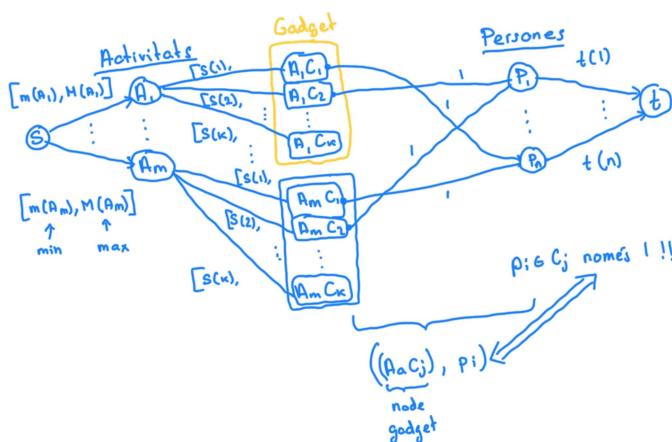
$s \rightarrow A_a \rightarrow (A_a, C_j) \rightarrow p_i \rightarrow t$

personna  $p_i$ : assisteix a l'activitat  
 $A_a$  com a membre del grup  $C_j$

La xarxa  $N$  té:

- nodes  $s, t,$

- arcs i capacetes



4.22. **(Discurs).** Davant el gran nombre de intervencions sobre la utilitat de l'ensenyament universitari, el rector de la UPC ha decidit encarregar a UPCnet un mecanisme per redactar aquest discursos.

UPCnet ha desenvolupat un model consistent en un graf dirigit  $G = (V, E, \omega)$ , on cada aresta és etiquetada amb una paraula, i un vèrtex distingit  $t \in V$ . Segons aquest model, un camí al graf que finalitza a  $t$  representa una frase amb sentit que es pot incloure al discurs. La frase es correspon amb la seqüència ordenada de mots que apareixen al camí, seguint l'ordre establert pel propi camí. Abans de preparar un discurs el rector decidirà els punts d'inici del seu discurs, tot indicant una seqüència de nodes del graf on han de començar les frases que el formaran.

Per tal d'eliminar redundàncies al discurs, l'equip rectoral aconsella que el discurs mai faci servir més d'una vegada una aresta del graf.

- (a) Dissenyeu un algorisme per escriure un discurs que acompleixi les especificacions del rector i segueixi els consells de l'equip rectoral. En cas que no fos possible, s'ha de presentar una versió reduïda del discurs on també s'especifiquin els nodes d'inici de frase que no s'han pogut completar.
- (b) Indiqueu com modificaríeu l'algorisme proposat a l'apartat previ si l'equip rectoral relaxa la condició prèvia de redundància i permet un nombre màxim  $c$  d'aparicions de cada aresta al discurs.

4.23. **(RACC).** El RACC està interessat a trobar un mecanisme per poder suggerir als seus clients rutes alternatives per fugir dels embussaments de trànsit al centre de Barcelona. El Servei Català de Trànsit (SCT) els hi ha proporcionat un model simplificat del centre de Barcelona, que està format per un graf  $G = (V, E)$  amb un vèrtex per cada cruïlla, i una aresta per cada parell de cruïlles connectades perquè hi ha un carrer que les uneix. A més coneixen el subconjunt de cruïlles  $B \subseteq V$  que donen accés a carrers que surten del centre de la ciutat.

Dels telèfons mòbils dels clients, el RACC pot obtenir informació sobre el conjunt  $C \subseteq V$  format per les cruïlles més rellevants als llocs on es troben els cotxes dels seus clients. L'SCT proporciona també el conjunt de trams de carrers  $E' \subseteq E$  amb un nivell baix de saturació de trànsit i el conjunt  $D \subseteq V$  de cruïlles crítiques que vol evitar.

El RACC vol determinar si és possible trobar un conjunt de camins disjunts (que no comparteixen arestes) que faci servir només trams amb baix nivell de saturació tals que, per cada cruïlla  $c \in C$  tinguem un camí que comença a  $c$  i que acaba a alguna de les cruïlles de  $B$ . A més cal que una cruïlla crítica (de  $D$ ) aparegui com a molt a 2 d'aquests camins.

Doneu un algorisme tan eficient com pugueu que, donats  $G$ ,  $E'$ ,  $B$ ,  $C$  i  $D$ , determini si és possible obtenir els camins requerits i que, en cas que ho sigui, retorna, per a cada  $c \in C$ , el camí que comença a  $c$  i acaba a alguna cruïlla de  $B$ .

4.24. La firma *Doctors on Call* té que resoldre el següent problema. Per a cadascun dels pròxims  $n$  dies, la firma ha determinat el nombre de doctors disponibles que requereix. Així al dia  $i$ -èsim, necessiten exactament  $p_i$  doctors. Hi han  $k$  doctors en total, i cadascú d'ells ha donat una llista amb els dies en que està disposat a treballar. Així el doctor  $j$  proporciona un conjunt  $L_j$  de dies. Doctors on Call vol, a partir d'aquesta informació, un procediment que permeti tornar a cada doctor  $j$  una llista definitiva de dies  $L'_j$  amb les propietats següents: (1) el conjunt  $\Delta_j = L'_j \setminus L_j$  té com a molt  $c$  dies; i (2) quan es considera tot el conjunt de llistes  $L'_1, \dots, L'_k$ , per a cada dia  $1 \leq i \leq n$ , hi han exactament  $p_i$  doctors que tenen el dia  $i$  a la seva llista definitiva. El paràmetre  $c$  reflecteix la tolerància de l'assignació i pot variar segons les circumstàncies. Per suposat, si tal solució no es possible, el sistema ha de (correctament) informar de que aquest és el cas.

- (a) Proporcioneu un algorisme de cost polinòmic en  $n$  i  $k$  que resolgui el problema per un valor de tolerància  $c$  donat.
- (b) Proporcioneu un algorisme de cost polinòmic que obtingui la tolerància mínima per que el problema tingui solució, assumint que  $p_i \leq k$  per  $1 \leq i \leq n$ .

- 4.25. **SafeTrans** es dedica al transport de mercaderies perilloses o de gran volum i/o pes per carretera. La companyia està analitzant la possibilitat d'acceptar un encarreg per traslladar els residus emmagatzemats a un dipòsit cap a un altre més segur. Per fer l'anàlisi de viabilitat el seu equip logístic ha definit un graf dirigit  $G = (V, E)$  que representa la xarxa de carreteres que permetria connectar els dos dipòsits  $s, t \in E$  dintre d'uns límits de quilometratge raonables.

Degut a la natura dels materials a transportar el trasllat s'ha de fer al llarg de com a màxim 5 dies durant la nit i a més s'ha de tenir en compte, en la mesura del possible, el nivell de seguretat que es pot assolir. Per això volen que l'equip informàtic els doni informació sobre la possibilitat de fer el trasllat tenint en compte diferents restriccions. En tots els escenaris, per raons de seguretat un tram de carretera només es pot fer servir com a molt una de les 5 nits.

**Escenari 1:** Si un tram de carretera es fa servir una nit, aquesta nit només hi pot circular per ell un camió carregat.

**Escenari 2:** Les condicions de l'escenari 1 es poden relaxar, per un subconjunt de trams de carretera suficientment aïllats. Suposant que  $S \subset E$  son els trams suficientment aïllats. En aquest escenari s'ha de garantir: Si un tram de carretera a  $S$  es fa servir una nit, aquesta nit poden circular un o dos camions carregats. Si un tram de carretera no és a  $S$  i es fa servir una nit, aquesta nit només hi pot circular un camió carregat.

**Escenari 3:** Una anàlisi alternativa de la densitat dels nuclis de població ha determinat que en la majoria dels casos la població està suficientment aïllada, però que hi han un cert subconjunt de nuclis urbans densament poblats. Suposant que  $D \subset V$  és el conjunt de nuclis urbans amb alta densitat de població. En aquest escenari s'ha de garantir que com a molt un camió carregat circuli una nit per  $d \in D$ . En contrapartida, si un tram de carretera es fa servir una nit, aquesta nit poden circular per ell fins a tres camions carregats.

Proporcioneu algorismes, per a cada escenari, què:

- Donats,  $G, S$  i  $D$  juntament amb  $s$  i  $t$ , determini un conjunt tan gran com sigui possible de rutes potencials per fer el transport en una nit.
- Donats,  $G, S, D, c, 1 \leq c \leq 5$ , i  $k$  juntament amb  $s$  i  $t$  determini si es possible seleccionar rutes per  $k$  camions i  $c$  dies, i en cas de que sigui possible proporcioni aquestes rutes, assegurant què: cada camió carregat fa com a molt un trajecte per nit; globalment es compleixen les restriccions imposades per l'escenari; i a més el nombre total de trasllats es més gran que  $\frac{2}{3}ck$ .

#### Solució:

- Resolveremos el problema como un problema de flujo máximo. Para cada escenario tendremos una red de flujo que modela el problema. Sobre esta red calcularemos el flujo máximo usando Ford Fulkerson. Una vez calculado un flujo con valor máximo consideraremos el grafo  $G'$  en el que solo aparecen las aristas con flujo positivo. En  $G'$  repetiremos el siguiente proceso: obtener un camino de  $s$  a  $t$ , este camino será una ruta; reducir el valor del flujo en una unidad en todas las aristas del camino; eliminar aquellas en las que el flujo sea 0.

En el escenario 1, tenemos que calcular caminos que no comparten aristas, tal y como hemos visto en clase basta con asignar capacidad 1 a todos los arcos en  $E$  y tomar  $s$  como fuente y  $t$  como sumidero.

En el escenario 2, los arcos en  $S$  pueden soportar dos caminos, les asignaremos capacidad 2 y a los que no están en  $S$  capacidad 1. Cualquier conjunto de rutas que cumpla las condiciones se puede convertir en un flujo en el que el valor del flujo en arcos de  $S$  es menor o igual que 2 y al revés siempre que el flujo tenga valores enteros. Por lo tanto un flujo entero con valor máximo nos proporciona una solución al problema en este escenario.

En el escenario 3, asignaremos a los arcos capacidad 3 y tal como hemos visto en clase para conseguir que los caminos no pasen dos veces por el mismo nodo en  $D$ , convertiremos el grafo en

un grafo  $G'$  donde cada nodo  $d \in D$  se remplaza por dos nodos  $d'$  y  $d''$ , los arcos que entran en  $d$  entraran en  $d'$ , los que salen lo harán de  $d''$  y añadimos un arco  $(d', d'')$  con capacidad 1.

En cualquiera de los tres casos el coste de calcular maxflow es  $O((n+m)m)$  ya que el valor del flujo máximo es como mucho  $3m$ . Con el mismo coste se pueden obtener la lista de rutas, realizando un BFS por cada unidad de flujo.

- (b) Notemos que los tramos de carretera solo se pueden utilizar en uno de los  $c$  días, por ello es necesario que el número total de rutas, calculado en el apartado a) sea  $\geq \frac{2}{3}ck$ .

Además necesitamos encontrar una partición de  $E$  en  $c$  conjuntos  $E_1, \dots, E_c$  que nos permita asignar  $\leq k$  rutas por día y suficientes rutas a lo largo de los  $c$  días. Para una partición  $E_1, \dots, E_c$ , resolveremos el problema de flujo máximo como en el apartado a) obteniendo  $c$  valores de flujo  $f_1, \dots, f_c$ , dónde  $f_i = \text{maxflow}(V, E_i, s, t, c)$ . La partición nos proporciona una solución si  $\sum_{i=1}^c \min k, f_i \geq \frac{2}{3}ck$ .

En el escenario 1 las rutas son totalmente independientes, ya que no comparten aristas y por lo tanto se pueden asignar a cualquiera de los días. Una vez resuelto a) asignamos las primeras  $k$  rutas al día 1, las siguientes  $k$  al día 2, hasta que acabemos con todas las rutas disponibles o llenemos los  $c$  días.

En los escenarios 2 y 3, recorreríamos todas las particiones posibles hasta encontrar una en la que las condiciones se cumplan o concluir que no hay tal solución.

En el escenario 1 el coste es como en el caso a)  $O((n + m)m)$ , pero podríamos reducirlo a  $O((n + m)ck)$  finalizando el algoritmo cuando el valor del flujo llegue al mínimo requerido. En los escenarios 2 y 3 tenemos que recorrer todas las particiones, este número es exponencial si  $c > 1$ , el algoritmo tiene coste exponencial, aún cuando el coste por partición es polinómico.

#### 4.26. (Vacunació).

La Generalitat ha de planificar els trasllats de les vacunes Covid des de l'aeroport del Prat als diferents centres de distribució de la vacuna a Catalunya. Per dur-ho a terme, ha creat un graf de distribució format per les localitats catalanes a on es poden emmagatzemar vacunes amb les condicions de conservació adients (incloent-hi el propi aeroport). Les connexions entre localitats garanteixen que el transport es pot portar a terme des d'una localitat fins a qualsevol altra, seguint una o més connexions, en temps suficientment curt per tal de no trencar la cadena del fred.

Per a cada localitat, fora del Prat, es coneixen la quantitat màxima de caixes de vacunes que es poden emmagatzemar de forma segura.

Per raons de seguretat, el nombre total de caixes de vacunes que poden circular per una localitat no pot superar la capacitat d'emmagatzematge de la localitat.

- (a) Dissenyeu un algorisme que, donats el graf de distribució, la quantitat de caixes que es pot emmagatzemar a cada localitat, les localitats que són centres de distribució i, per a cada centre de distribució, el nombre de caixes de vacunes que es volen traslladar des del magatzem de l'aeroport a ell, determini si és possible fer el trasllat respectant les restriccions descrites abans.  
Podeu suposar que al magatzem del Prat hi ha més caixes de vacunes de les que es volen distribuir.
- (b) En cas que el trasllat no es pugui portar a terme, esteneu l'algorisme precedent per tal d'identificar un conjunt de localitats on un increment de seva capacitat d'emmagatzematge garantiria poder portar a terme el trasllat.

#### Una solución:

- (a) Podemos resolver el problema como un problema de circulación con demandas en una red adecuada. La limitación de transporte está en la cantidad de cajas de vacunas que pueden atravesar una localidad.

Sea  $G = (V, E)$  el grafo que nos proporcionan y  $a(u)$ ,  $u \in V$ , la cantidad de cajas que se pueden almacenar la localidad  $u$ . Suponemos que  $s \in V$  es el nodo que representa el almacén del Prat. Si  $D \subset V - \{s\}$  es el conjunto de centros de distribución, se nos dará  $b(w) > 0$ ,  $w \in D$ , el número de cajas de vacunas que queremos trasladar desde el Prat al centro de distribución  $w$ .

Asumimos que el grafo que nos dan es dirigido, si no lo fuese, tal y como hemos visto en clase se convertiría en dirigido poniendo los arcos  $(u, v)$  y  $(v, u)$  para cada arista  $\{u, v\}$  del grafo no dirigido.

La red  $\mathcal{N}$  tiene

- Nodos:  $V$  y  $V'$ , siendo  $V'$  una copia de los nodos en  $V$ . Si  $u \in V$ , entonces  $u'$  representa su copia en  $V'$ .
- Aristas y capacidades:

$$\begin{aligned} \{(u, u') \mid u \in V\} &\quad \text{capacidad } a(u) \\ \{(u', v) \mid (u, v) \in E\} &\quad \text{capacidad } \infty \end{aligned}$$

- Demandas:

$$\begin{aligned} u \in D, d(u) &= b(u) \\ d(s) &= - \sum_{u \in D} b(u) \\ u \notin D, u \neq s, d(u) &= 0 \end{aligned}$$

La duplicación de los nodos y la capacidad de la arista que conecta un nodo con su copia garantizan que, en un flujo entero, nunca pasan por un nodo más cajas de las que es posible almacenar en él. Teniendo en cuenta que el aeropuerto tiene suficientes cajas para satisfacer la demanda, el problema de circulación tiene solución si y solo si podemos realizar el transporte requerido ya que las restricciones solicitadas están garantizadas por la capacidad de las aristas.

Para analizar el coste, asumo que  $G$  tiene  $n$  vértices y  $m$  aristas. Así  $\mathcal{N}$  tiene  $2n$  vértices y  $n+m$  aristas. Como no tenemos cotas en el valor de las demandas y  $-d(s)$  puede ser grande, el coste del algoritmo viene dado por el análisis de EK con coste  $O(NM^2) = O(n(n+m)^2)$ .

- (b) Si la circulación no existe, lo que podemos hacer es considerar la red de flujo utilizada para resolver el problema de circulación y el flujo con valor máximo que hemos calculado en dicha red. Utilizando el grafo residual, obtenemos un corte con capacidad mínima. Como ninguna de las aristas del corte puede tener capacidad  $\infty$ , ya que no permite el transportar todas las cajas requeridas, cada arista de este corte identifica un vértice.

Miraríamos en cuanto se incrementa el valor del flujo asignando capacidad  $\infty$  a las aristas del corte. Si con esto no es suficiente para que el problema de circulación tenga solución. Repetiremos el proceso, con la nuevo red y el nuevo flujo máximo, tantas veces como sea necesario. El algoritmo acabará en el momento en que la circulación buscada exista.

El conjunto de aristas  $(u, u')$  con capacidad  $\infty$ , nos proporciona el conjunto de nodos pedidos. Observemos que en cada paso el corte con capacidad mínima no puede contener ninguna de las aristas a las que hemos asignado capacidad  $\infty$ , por lo que como mucho repetiremos este proceso  $O(n)$  veces, cada una de ellas con el coste de EK  $O(n(n+m)^2)$ . Así el coste total del algoritmo es  $O(n^2(n+m)^2)$ .