PONTIFICIA UNIVERSIDAD CATÓLICA DE CHILE
ESCUELA DE INGENIERÍA
DPTO. DE INGENIERÍA DE TRANSPORTE Y LOGÍSTICA

Curso: ICT2233
Semestre: 2023-01
Profesor: Mathias Klapp

INTERROGACIÓN 2
FLUJO EN REDES - ICT2233

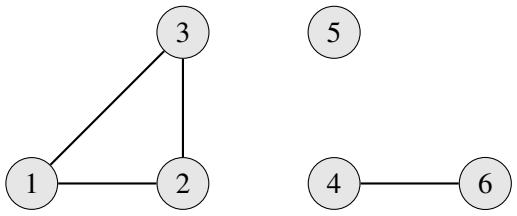
TIEMPO: 120 MINUTOS

- *La prueba tiene 60 puntos distribuidos entre todas sus preguntas. Sea estratégico, hay algunas preguntas más difíciles que otras.*
- *¡Coloque nombre en el cuadernillo y no lo separe!* ***Se penalizará con 4 puntos a quien no siga esta instrucción****.*
- *La prueba es sin apuntes y/o equipos electrónicos. Se permite resumen de una hoja escrito a mano.*

¡Éxito!

Pregunta 1 (5 puntos)

Considere un grafo no dirigido $G = (N, A)$, donde $m = |A|$. Una componente conectada de G es un subgrafo inducido por un subconjunto de N tal que cada uno de sus nodos está conectado con todo nodo del subconjunto y desconectado del resto de los nodos del grafo. Por ejemplo, el siguiente grafo posee 3 componentes conectadas dadas por los subgrafos inducidos por $N_1 = \{1,2,3\}$, $N_2 = \{4,6\}$ y $N_3 = \{5\}$:



Diseñe (en palabras o pseudo-código) un algoritmo capaz de contar la cantidad de componentes conectadas en un grafo G cualquiera que ejecute en tiempo $\mathcal{O}(m)$ o menor.

NOTA: En el diseño de su algoritmo puede hacer uso de algoritmos vistos en clase sin detallarlos, por ejemplo: Bellman-Ford, DFS, Dijkstra, Ordenamiento topológico, Prim, etc.

Respuesta: Iniciar contador $k = 0$. Luego, ejecutar un DFS (o BFS) no dirigido desde un nodo (escogido al azar) hacia el resto. Todo lo que conecte es la primera componente conectada. Luego, elimine esta componente (arcos y nodos), haga $k+ = 1$ y repita desde otro nodo hasta consumir todo el grafo. Finalmente, retorne k (el conteo de componentes conectadas).

Pregunta 2 (20 puntos)

Suponga que usted debe coordinar la ejecución de un proyecto formado por un conjunto finito de tareas $T := \{1,\ldots,n\}$. Por ejemplo, el proyecto podría ser construir una casa y sus tareas serían: "construir ciminetos", "construir murallas", "instalar puertas", "instalar ventanas", "pintar casa", etc.

Usted debe **agendar el proyecto**, es decir, definir cuando comienza a ejecutarse cada tarea $i \in T$ (modelado en la variable $x_i \geq 0$) cumpliendo las siguientes restricciones:

i. Una tarea $i \in T$ inicia en tiempo no-negativo, es decir, $x_i \geq 0$.

ii. Una tarea $i \in T$ se ejecuta en $t_i \geq 0$ horas, es decir, termina en $x_i + t_i$.

iii. El proyecto termina después de terminar todas sus tareas, es decir, su duración es $D = \max_{i \in T}\{x_i + t_i\}$.

iv. Finalmente, cada tarea $i \in T$ posee un subconjunto de tareas $P_i \subseteq T$ prerrequisito. Es decir, cada tarea $j \in P_i$ debe terminarse antes de iniciar la tarea i (matemáticamente, se exige que $x_i \geq x_j + t_j$ para todo $j \in P_i$ y todo $i \in T$). Por ejemplo, la tarea "construir cimientos" es prerrequisito de "construir murallas". Una tarea $i \in T$ sin prerrequisitos (es decir, $P_i = \emptyset$) se denomina **primaria**.

Responda a cada una de las siguientes preguntas:

(a) (4p) Pruebe la siguiente afirmación: "Para poder agendar el proyecto (es decir, definir valores factibles de cada $x_i, \forall i \in T$) es necesario que exista al menos una tarea primaria en T." **Respuesta:** Para probarla, supongamos que no es correcta (el proyecto se puede agendar y no hay tareas primarias). En este caso se puede construir una secuencia infinita de tareas $(i_1, i_2, i_2, \ldots)$ tal que $i_{k+1} = P_{i_k}$ para todo $k \geq 1$. Por lo tanto, hay dos posibilidades:

- ◇ Se repiten tareas en la secuencia y existe un ciclo de prerrequisitos que hace imposible ejecutar el proyecto ($\Rightarrow\Leftarrow$).
- ◇ No se repiten y hay infinitas tareas ($\Rightarrow\Leftarrow$).

Por lo tanto, es contradictorio y no se puede ejecutar el proyecto sin tareas primarias.

(b) (2p) En un ejemplo, muestre que la existencia de tareas primarias no garantiza poder agendar el proyecto.

Respuesta: Puede haber un ciclo de prerrequisitos igual. Por ejemplo, supongamos 3 tareas ($T := \{1,2,3\}$) tal que $P_1 = \emptyset$, $P_2 = \{3\}$ y $P_3 = \{2\}$. Es imposible agendar este proyecto.

(c) (2p) ¿Qué condición es necesaria y suficiente para poder agendar el proyecto? ¿Cómo validaría rápidamente que se cumple la condición?

Respuesta: Se debe exigir que no existan ciclos de prerrequisitos Para ellos se puede revisar la existencia de ciclos en un grafo donde los nodos son las tareas y se agregan un arco (j,i) por cada prerrequisito $j \in P_i$ en cada tarea $i \in T$.. Se puede verificar buscando un OT en $\mathcal{O}(m)$. Si no se puede asignar, entonces hay dependencia cíclica.

(d) (5p) Represente el problema asociado a agendar el proyecto de mínima duración D como una instancia particular del Problema de Ruta Mínima (PRM) desde un nodo origen hacia todo el resto.

NOTA: No necesita escribir el modelo matemático. Solo determine cada elemento del PRM (nodos, arcos, costos) en función de los datos entregados.

¿Que representa la distancia de ruta mínima a cada nodo? ¿Qué algoritmo recomendaría para resolver eficientemente este problema?

Respuesta: Formemos un grafo $G := (N,A)$ con costos c_a en cada uno de sus arcos $a \in A$:

◇ Nodos: El conjunto de nodos será $N = T \cup \{0\}$, donde 0 será nuestro nodo origen.

◇ Arcos: Se necesitan dos clases de arcos en A.

- Arcos $(0,i)$ desde 0 a cada nodo $i \in T$ a costo t_i.
- Arcos (j,i) desde cada $j \in P_i$ a cada nodo $i \in T$ a costo t_i.

El costo d_i de ruta máxima de 0 hacia cada tarea $i \in T$ indica el menor tiempo de término de la tarea i, pues identifica la cadena de tareas más extensa en tiempo requerida para terminar i, luego $x_i^* = d_i - t_i$ y $D = \max_{i \in T} d_i$. El grafo será acíclico si existe agenda factible (si no hay un ciclo de prerrequisitos) y, por lo tanto, es equivalente a un PRM acíclico cuando se multiplica los costos en arcos por -1. Se puede resolver con un algoritmo de ruta mínima por order topológico.

NOTA: Hay varias formas de modelar esta red.

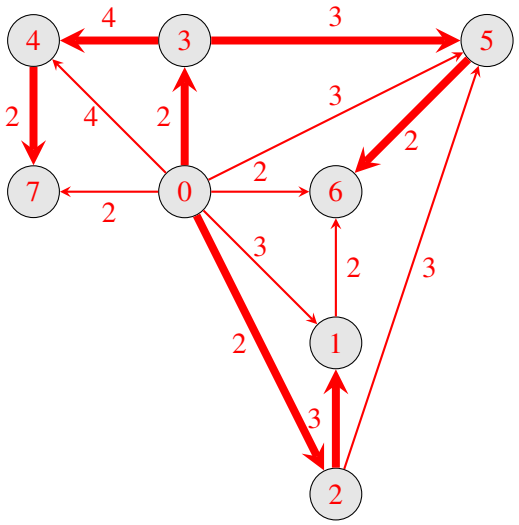
(e) (5p) Considere la siguiente instancia del problema de agendamiento de proyecto. Cada tarea detalla sus prerrequisitos y su tiempo de ejecución:

tarea (i)	Prerequisitos (P_i)	horas de ejecución (t_i)
1	2	3
2	no hay	2
3	no hay	2
4	3	4
5	2,3	3
6	1,5	2
7	4	2

Determine la mínima duración posible de este proyecto resolviendo el PRM planteado en la pregunta anterior. ¿Cuándo debe iniciar cada tarea?

Respuesta: En el siguiente grafo se modela el problema planteado y se detalla el árbol de rutas máximas desde 0. La duración mínima del proyecto es 8. Los tiempos de inicio y termino por tarea son:

	1	2	3	4	5	6	7	D
inicio	2	0	0	2	2	5	6	0
termino	5	2	2	6	5	7	8	8



Pregunta 3 (7 puntos)

Considere un grafo dirigido (N,A) que detalla un costo $c_a \in \mathbb{Z}_+$ en cada arco $a \in A$ y un nodo fuente $s \in N$ desde el cual se desea calcular una ruta de mínimo costo total hacia otro nodo sumidero $t \in N$.

Previamente, se ejecutó DFS y BFS desde s y se identificó dos rutas hacia t de costo 15 y 17, respectivamente.

Ahora, suponga que usted está ejecutando el algoritmo Dijkstra (específicamente, su versión A^*) y va a comenzar una iteración intermedia desde el nodo $i \in N$ (último nodo cerrado) hacia los nodos que quedan abiertos $j \in \{1,2,3,4,5\}$. Esta es la información disponible:

- El costo de sub-ruta desde s hacia i es $d_i = 8$.
- Para cada nodo abierto $j \in \{1,2,3,4,5\}$, usted tiene el costo $c_{i,j}$, el costo actual de sub-ruta d_j desde el nodo s hacia j y, además, cuenta con una subestimación del costo de ruta mínima desde j hacia t dado por u_j. La siguiente tabla resume esta información:

nodo (j)	$c_{i,j}$	d_j	u_j
1	2	11	4
2	0	16	8
3	4	16	3
4	2	9	0
5	3	15	3

Ejecute una iteración del algoritmo A* y actualice la información de distancia d_j de los nodos abiertos. ¿Cuál es el próximo nodo a cerrar?

Respuesta: Se tiene un incumbente de $I = 15$. (1 punto por aclarar esto).

Para actualizar j se requiere dos criterios:

- Mejorar d_j, es decir, $d_i + c_{ij} < d_j$
- Tener chance de ganarle al incumbente: $d_i + c_{ij} + u_j < I$

Analicemos cada caso (1 punto cada uno):

nodo 1: En este caso $d_i + c_{i1} = 10 < d_1 = 11$ y $d_i + c_{i1} + u_1 = 14 < I$. Luego, se debiese actualizar $d_1 = 10$.

nodo 2: Se tiene que $d_i + c_{i2} = 8 < d_2 = 16$, pero $d_i + c_{i2} + u_2 = 16 > I$. Luego, no se debiese actualizar d_2, pues no se utilizará nunca en una ruta de mínimo costo.

nodo 3: Se tiene que $d_i + c_{i3} = 12 < d_3 = 16$, pero $d_i + c_{i3} + u_3 = 15 = I$. Luego, no se debiese actualizar d_3, pues ya tenemos una ruta de costo 15 y no hay forma de bajar ese costo via 3.

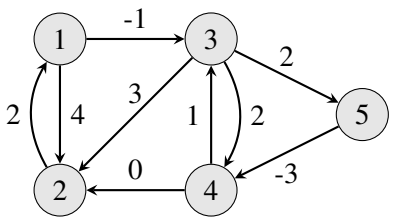
nodo 4: Se tiene que $d_i + c_{i4} = 10 > d_4 = 9$, por lo que no conviene actualizar.

nodo 5: Se tiene que $d_i + c_{i5} = 11 < d_5 = 15$ y $d_i + c_{i5} + u_5 = 14 < I$. Luego, se debiese actualizar $d_5 = 11$.

Finalmente, el próximo nodo a cerrar es 4 (1 punto).

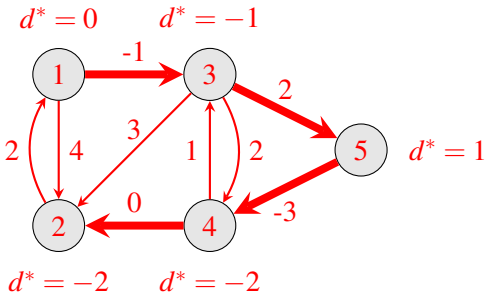
Pregunta 4 (15 puntos)

Considere el siguiente grafo $G = (N, A)$ dirigido que detalla un costo c_a en cada arco $a \in A$:



(a) (6p) Identifique un árbol de rutas mínimas desde el nodo 1 hacia el resto de los nodos en G mediante un algoritmo de rutas mínimas válido para esta instancia (diferente a simplex de redes). Señale qué algoritmo utilizó y detalle su procedimiento.

Respuesta: Este es un árbol de rutas mínimas desde 1:

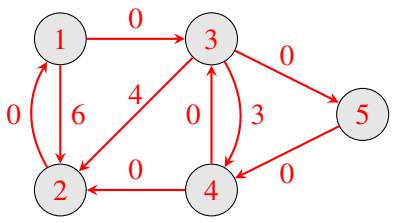


Se puede obtener mediante un algoritmo del tipo Label Correcting, *e.g.*, Bellman-Ford, DÉsopo-Pape u otro.

Debe detallar ejecución del algoritmo si aspira al máximo puntaje (3p por la solución correcta y 3p por la ejecución del algoritmo).

(b) (4p) Diseñe un nuevo grafo con los mismos nodos y arcos, pero que tenga costos no negativos en cada arco y que posea el mismo conjunto de rutas de mínimo costo entre cada par que G. Explique.

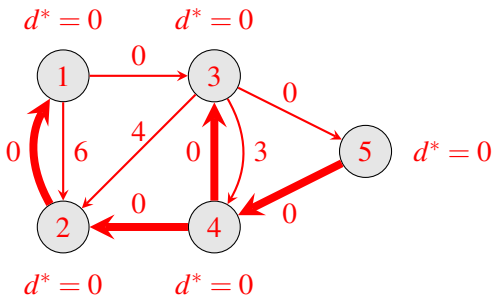
Respuesta: En clase vimos que una forma es definir $c'_{ij} = c_{ij} + d^*_i - d^*_j$. El resultado es:



(c) (5p) Ejecute el algoritmo de Dijkstra en el nuevo grafo desde el nodo 5 hacia el resto de los nodos. Luego, valide la optimalidad de estas rutas en el grafo original verificando condiciones que la garanticen.

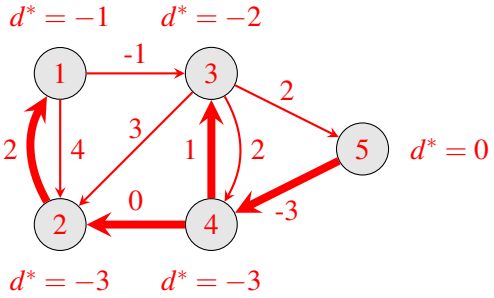
Respuesta:

Después de ejecutar Dijkstra desde 5 (orden de cierre: 5,4,2 o 3, 1) se obtiene:



(3 puntos hasta acá)

En la red de costos originales, este árbol es:

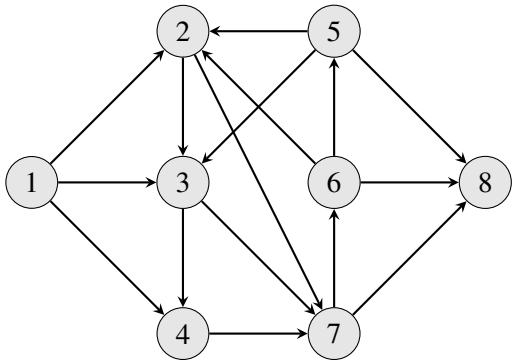


Para verificar su optmalidad revisaremos la no-negatividad de sus costos reducidos en arcos fuera de base (2 puntos por esto).

- $r_{12} = 4 + (-1) - (-3) = 6$
- $r_{13} = -1 + (-1) - (-2) = 0$
- $r_{32} = 3 + (-2) - (-3) = 4$
- $r_{34} = 3 + (-2) - (-3) = 4$
- $r_{35} = 3 + (-2) - (0) = 1$

Pregunta 5 (8 puntos)

Considere el siguiente grafo dirigido (N,A) que representa a una red de caminos existente entre 8 localidades.



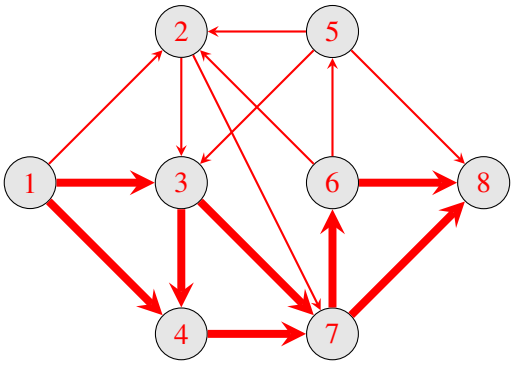
Suponga que usted está a cargo de fiscalizar el tránsito y sabe que algunos conductores viajan en su vehículo sin placa patente desde 1 hacia 8. Para identificar y detener a los infractores, puede asignar diariamente a un grupo de fiscalizadores en algunos de los caminos en la red. Un fiscalizador asignado a un camino es capaz de multar a todo conductor que transita sin patente por dicho camino. Responda a las siguientes preguntas:

(a) (6p) ¿Cuál es el mínimo número de fiscalizadores requerido para multar a todo infractor que va desde 1 hacia 8? Modele el problema como uno de flujo máximo e identifique el flujo (x) y el corte (Q) que certifican el valor de su respuesta.

Respuesta: Es un problema de mínimo corte con todo arco de capacidad 1 (*i.e.*, problema de rutas disjuntas en arcos.).

El corte de menor capacidad es $Q = \{1,2,3,4,7\}$ que posee capacidad 2. Esto se certifica con las siguientes dos rutas disjuntas en arcos:

En el grafo a continuación se detallan dos rutas:



(b) (2p) ¿Cuál sería su respuesta anterior si también puede asignar fiscalizadores a intersecciones de caminos (*i.e.*, nodos)? Explique.

Respuesta: Si, pues se transformaría en un problema de rutas disjuntas en nodos (y solo hay una ruta disjunta en nodos). En este caso, bastaría con fiscalizar en el nodo 7.

Pregunta 6 (5 puntos)

Considere un problema de flujo máximo (PFM) sobre un grafo dirigido $G = (N, A)$ con $n = |N|$, $m = |A|$ y capacidades en arcos $u_a \in \mathbb{Z}_+$, $\forall a \in A$. Suponga que el valor de flujo máximo desde un nodo $s \in N$ hacia otro nodo $t \in N$ es igual a $F^* < \infty$.

Llame **arco crítico hacia abajo** a todo arco $a \in A$ tal que al reducir su capacidad u_a en una unidad genera una reducción del valor de flujo F^* en una unidad.

Proponga un algoritmo que identifique al conjunto de arcos críticos hacia abajo en G con menor complejidad computacional que resolver el problema de flujo máximo m veces. Justifique.

Respuesta: Para que un arco (i, j) sea crítico hacia abajo se deben cumplir dos condiciones:

- **(a)** Debe tener flujo a capacidad ($x_{i,j} = u_{i,j}$). Si no, se puede reducir $u_{i,j}$ y no se altera su flujo (ni en consecuencia la solución que produce un valor F^*).
- **(b)** No puede existir una ruta desde i hasta j en la red residual (de lo contrario, implica que existe una alternativa al utilizar el arco (i, j) cuando este reduzca su capacidad).

Luego, para verificar esto se puede revisar por DFS que no exista una ruta desde i a j para cada arco (i, j) que esté a capacidad. En el peor caso corre en $\mathcal{O}(n \cdot m)$ operaciones (n algoritmos de búsqueda).