UNIVERSIDAD TORCUATO DI TELLA

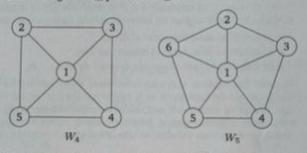
Tecnología Digital V: Diseño de algoritmos
Licenciatura en Tecnología Digital
Primer Semestre, 2024

Primer parcial

Nombre y Apellido: Camilo Daniel Suárez Canónico
El parcial se aprueba con 60/100. Resolver cada ejercicio en hojas separadas.
Justifique **todas** sus respuestas.

1. Recordemos que un coloreo de un grafo $G = (V, E)$ es una asignación $c : V \to \{1, \ldots, n\}$ de colores a los vértices de G de modo tal que $c(i) \neq c(j)$ para toda arista $ij \in E$. El *número cromático* de G es la mínima cantidad de colores para la cual existe un coloreo de G.

   a) (15 puntos) Para $n \geq 3$, el *grafo rueda* W_n es un grafo compuesto por un ciclo de n vértices más un vértice universal (es decir, un nuevo vértice vecino de todos los vértices del ciclo). Por ejemplo, la siguiente figura muestra los grafos W_4 y W_5. ¿Cuál es el número cromático del grafo W_n, para todo $n \geq 3$?



   b) (5 puntos) Mostrar un grafo que tenga número cromático igual a 4 pero que no tenga ningún subgrafo completo con cuatro vértices (es decir, mostrar un grafo G con $\chi(G) = 4$ pero $\omega(G) < 4$).

2. Consideramos el problema de la mochila 0-1 *con grafo de conflictos*, KP-01-GC, definido de la siguiente manera. Sea $N = \{1, \ldots, n\}$ un conjunto de objetos, con $p_i \in \mathbb{Z}_+$ el peso y $b_i \in \mathbb{Z}_+$ el beneficio asociado al producto $i \in N$. El peso máximo de la mochila está dado por $C \in \mathbb{Z}_+$. Adicionalmente, se tiene como input un *grafo de conflictos* $G = (N, E)$, con un vértice por cada objeto y donde una arista $\{i, j\} \in E$ indica que los objetos i y j son *incompatibles* y no pueden estar ambos simultáneamente en la mochila.

   A modo de ejemplo, consideremos una instancia con $n = 4$ productos, con pesos $p_i = 2$ para $i = 1, \ldots, n$, $C = 6$ y el grafo de conflictos $G = (N, E)$, con $N = \{1, 2, 3, 4\}$ y $E = \{(1, 2), (1, 3), (3, 4)\}$ (por simplicidad, no especificamos los beneficios). La solución $S = \{1, 3, 4\}$ satisface la capacidad de la mochila, pero los productos 1 y 3 son incompatibles y entonces esta solución no es factible. La solución $S = \{2, 4\}$ es factible ya que cumple ambas restricciones.

   a) (20 puntos) Dar un algoritmo de fuerza bruta para el problema, incluyendo el pseudocódigo de aquellas funciones auxiliares que sean relevantes.

   b) (10 puntos) Proponer una poda por factibilidad que aproveche la presencia del grafo de conflictos.

   c) (10 puntos) Un colega propone un algoritmo de programación dinámica usando un enfoque top-down, con la siguiente estructura para aplicar *memoization*:

"podríamos usar una matriz de doble entrada `memo`, donde `memo[k][c]` indica el valor óptimo del problema con los primeros $\{1, \ldots, k\}$ elementos y capacidad remanente c."
Decidir si conviene o no utilizar esta propuesta. Justificar apropiadamente.

3. Estamos a cargo del departamento de recursos humanos de una empresa, y tenemos que otorgarle días de vacaciones adeudados a un conjunto de empleados. Llamamos $E = \{1, \ldots, n\}$ al conjunto de estos empleados y llamamos $D = \{1, \ldots, m\}$ a los días hábiles a partir del próximo lunes. Cada empleado $i \in E$ pidió un período de vacaciones, comenzando en el día $s_i \in D$ y terminando en el día $e_i \in D$ (con $s_i < e_i$). Dado que todos estos empleados están participando en un mismo proyecto, no podemos otorgar vacaciones a más de un empleado al mismo tiempo (con lo cual es posible que no podamos otorgarle vacaciones a todos).

   a) (15 puntos) El problema que queremos resolver es maximizar la cantidad de empleados a los cuales se les otorgan vacaciones en el período que solicitaron, teniendo en cuenta que no puede haber dos empleados de vacaciones al mismo tiempo. Los empleados a los que no se les otorga vacaciones tendrán que hacer un nuevo pedido más adelante. Mostrar cómo se puede modelar este problema como un problema de camino máximo en una red convenientemente construida.

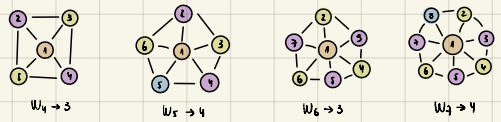
   b) (25 puntos) Consideremos ahora la siguiente variación del problema anterior. No es necesario asignar al empleado $i \in E$ el intervalo que va desde el día s_i hasta el día t_i, sino que se puede asignar un intervalo menor, siempre que cumpla estas condiciones:

   - El intervalo debe comenzar el día s_i.
   - El intervalo debe durar al menos dos días.
   - El intervalo no debe terminar después del día s_i. e_i

   El problema consiste ahora en determinar a qué empleados se otorgan vacaciones y qué intervalo de vacaciones se le asigna a cada uno, con el objetivo de maximizar los días totales de vacaciones asignados. Nuevamente, no puede haber dos empleados de vacaciones al mismo tiempo. Mostrar cómo se puede modelar este problema como un problema de camino máximo en una red convenientemente construida.

Ejercicio 1

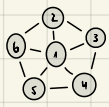
Número cromático de W_n, para todo $n \geq 3$?



Si n es par usaré siempre 3 colores como mínimo. 1 si o sí irá para el vértice universal, otro color irá para n/2 vértices, y el último para los restantes n/2 vértices. Al ser par, siempre habrán igual cant. de vértices con el color 2 que con el 3, y se los podrá "intercalar" para respetar el coloreo.

Ahora, si n es impar el número cromático será 4. 1 color nuevamente es solo para el vértice universal. Otro irá para $\frac{(n-1)}{2}$ vértices. El tercero irá para otros $\frac{(n-1)}{2}$ vértices. Nuevamente, los puedo intercalar, pero esta vez quedará uno suelto que se conecta con vértices de los 3 colores, por lo que no puede tomar ninguno de los 3. Por eso es que se suma el 4to color.

b) Grafo con número crom = 4 pero con clique < 4



Ejercicio 2

N = {1, ..., n} objetos

p_i positivo el peso del objeto $i \in N$

b_i positivo el beneficio del objeto $i \in N$

C positivo el peso máx de la mochila

Grafo de conflictos → si 2 vértices son vecinos, son incompatibles.

Solución: lista de objetos con 1s y 0s dependiendo si meto el obj. o no

```
FB (S, n)
if n == 0:
    if esfactible(S) ∧ beneficio(S) > beneficio(best):
        best ← S
else:
    S' ← S + [1]
    FB ← (S, n-1)
    S' ← S + [0]
    FB ← (S, n)
```

best es global, por lo que se va modificando.

esfactible chequea que en primer lugar, los objetos de S sean compatibles todos con todos. Luego, que la suma de los pesos de solución sean menor a la cap. máx.

```
esfactible (S):
    res = false
    if (Todos compatibles && Σ_{i=0}^{len(S)} bi ≤ C)
        res = true
    return res
```

b) Poda por factibilidad

```
FB (S, n)
if n == 0 :
    if esfactible(S) ∧ beneficio(S) > beneficio(best) :
        best ← S
else if esfactible :
    S' ← S + [1]
    FB ← (S, n-1)
    S' ← S + [0]
    FB ← (S, n)
```

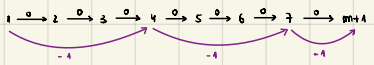
c) No, luego amplío.

Ejercicio 3

$E = \{1, \dots, n\}$ empleados, cada uno pide vacaciones si $a_{ei} \in D$

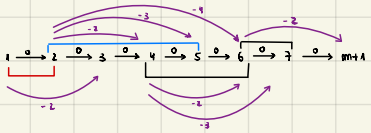
$D = \{1, \dots, m\}$ días hábiles

a) Parecido al de vacanza.



En este ejemplo $m=7$. Tengo un empleado que se pide del día 1 (incluido) al día 4 (sin incluir), otro que se pide del día 4 al día 7 (sin incluir), y el último que se pide el día 7. Para cumplir la dinámica, ya que las aristas no incluyen el último del grupo pero sí al primero, se agregó un día ficticio extra para que no se rompa si alguien quiere pedirse el último día y nada más, así es que en ese "grupo" incluye al día 7 pero no al m+1. Las aristas pueden tomar peso 0 ya que pueden haber días que nadie toma, así el grafo es conexo. Por otro lado, agrego las aristas -1 para buscar camino máximo usando algún algoritmo como Bellman-Ford con pesos negativos.

b)



Acá modelo las aristas como cant. de días que se le pueden dar a tal empleado que pide x días. Si el empleado se pide + de 2 días (por ej 4), se le puede dar 4, 3 ó 2. Si ejecuto un algoritmo de camino mínimo con pesos negativos voy a maximizar la cant. de días, eligiendo de todas las opciones de días para dar, la mayor.

representa persona que pide 2 días

persona que se pide 4 días, pero se le pueden dar 4, 3 ó 2.