Teoría de Algoritmos I (75.29, 95.06) – Curso Buchwald - Genender

Examen parcial – 10/11/2023

1. Somos ayudantes del gran ladrón *el Lunático*, que está pensando en su próximo atraco. Decidió en este caso robar toda una calle en un barrio privado, que tiene la particularidad de ser circular. Gracias a los trabajos de inteligencia realizados, sabemos cuánto se puede obtener por robar en cada casa. Podemos enumerar a la primer casa como la casa 0, de la cual podríamos obtener g_0, la casa a su derecha es la 1, que nos daría g_1, y así hasta llegar a la casa $n-1$, que nos daría g_{n-1}. Como la calle es circular, la casa 0 y la $n-1$ **son vecinas**. El problema con el que cuenta *el Lunático* es que sabe de experiencias anteriores que, si roba en una casa, los vecinos directos se enterarían muy rápido. No le daría tiempo a luego intentar robarles a ellos. Es decir, para robar una casa debe prescindir de robarle a sus vecinos directos. *El Lunático* nos encarga saber cuáles casas debería atracar y cuál sería la ganancia máxima obtenible. Dado que nosotros nos llevamos un porcentaje de dicha ganancia, vamos a buscar el óptimo a este problema. Implementar un algoritmo que, por **programación dinámica**, obtenga la ganancia óptima, así como cuáles casas habría que robar, a partir de recibir un arreglo de las ganancias obtenibles. Para esto, escribir y describir la ecuación de recurrencia que correspondiente. Indicar y justificar la complejidad del algoritmo propuesto.

2. Tenés una colección de n libros con diferentes espesores, que pueden estar entre 1 y n (valores no necesariamente enteros). Tu objetivo es guardar esos libros en la menor cantidad de cajas. Todas las cajas que disponés son de la misma capacidad L (se asegura que $L \geq n$). Obviamente, no podés partir un libro para que vaya en múltiples cajas, pero sí podés poner múltiples libros en una misma caja, siempre y cuando no superen esa capacidad L. Implementar un algoritmo Greedy que obtenga la mínima cantidad de cajas a utilizar. Indicar y justificar la complejidad del algoritmo implementado. Justificar por qué se trata de un algoritmo greedy (no dar una respuesta genérica, sino aplicada a tu algoritmo). ¿El algoritmo propuesto encuentra siempre la solución óptima? Justificar.

3. Un set dominante (Dominating Set) de un grafo G es un subconjunto D de vértices de G, tal que todo vértice de G pertenece a D o es adyacente a un vértice en D. El problema de decisión del set dominante implica, dado un grafo G y un número k, determinar si existe un set dominante de a lo sumo tamaño k.

   Demostrar que el Dominating Set Problem es un problema NP-Completo. Ayuda: recomendamos recordar Vertex Cover, que puede ser útil para esto.

4. Implementar un algoritmo que reciba un Grafo y un número k y devuelva un dominating set de dicho grafo de a lo sumo k vértices (si existe).

5. Hacer un seguimiento de obtener el flujo máximo en la red de transporte del dorso, realizando las modificaciones previas que fueran necesarias. Luego, definir cuáles son los dos conjuntos del corte mínimo en dicha red.

Teoría de Algoritmos I (75.29, 95.06) – Curso Buchwald - Genender

Examen parcial – 10/11/2023

1. Somos ayudantes del gran ladrón *el Lunático*, que está pensando en su próximo atraco. Decidió en este caso robar toda una calle en un barrio privado, que tiene la particularidad de ser circular. Gracias a los trabajos de inteligencia realizados, sabemos cuánto se puede obtener por robar en cada casa. Podemos enumerar a la primer casa como la casa 0, de la cual podríamos obtener g_0, la casa a su derecha es la 1, que nos daría g_1, y así hasta llegar a la casa $n-1$, que nos daría g_{n-1}. Como la calle es circular, la casa 0 y la $n-1$ **son vecinas**. El problema con el que cuenta *el Lunático* es que sabe de experiencias anteriores que, si roba en una casa, los vecinos directos se enterarían muy rápido. No le daría tiempo a luego intentar robarles a ellos. Es decir, para robar una casa debe prescindir de robarle a sus vecinos directos. *El Lunático* nos encarga saber cuáles casas debería atracar y cuál sería la ganancia máxima obtenible. Dado que nosotros nos llevamos un porcentaje de dicha ganancia, vamos a buscar el óptimo a este problema. Implementar un algoritmo que, por **programación dinámica**, obtenga la ganancia óptima, así como cuáles casas habría que robar, a partir de recibir un arreglo de las ganancias obtenibles. Para esto, escribir y describir la ecuación de recurrencia que correspondiente. Indicar y justificar la complejidad del algoritmo propuesto.

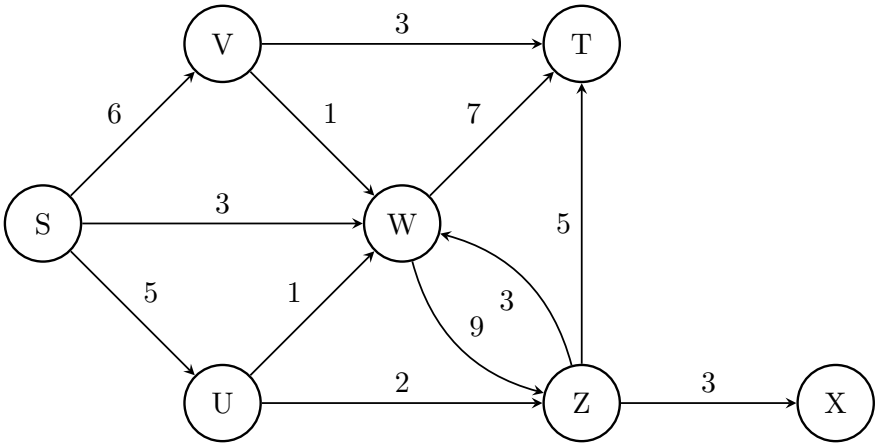
2. Tenés una colección de n libros con diferentes espesores, que pueden estar entre 1 y n (valores no necesariamente enteros). Tu objetivo es guardar esos libros en la menor cantidad de cajas. Todas las cajas que disponés son de la misma capacidad L (se asegura que $L \geq n$). Obviamente, no podés partir un libro para que vaya en múltiples cajas, pero sí podés poner múltiples libros en una misma caja, siempre y cuando no superen esa capacidad L. Implementar un algoritmo Greedy que obtenga la mínima cantidad de cajas a utilizar. Indicar y justificar la complejidad del algoritmo implementado. Justificar por qué se trata de un algoritmo greedy (no dar una respuesta genérica, sino aplicada a tu algoritmo). ¿El algoritmo propuesto encuentra siempre la solución óptima? Justificar.

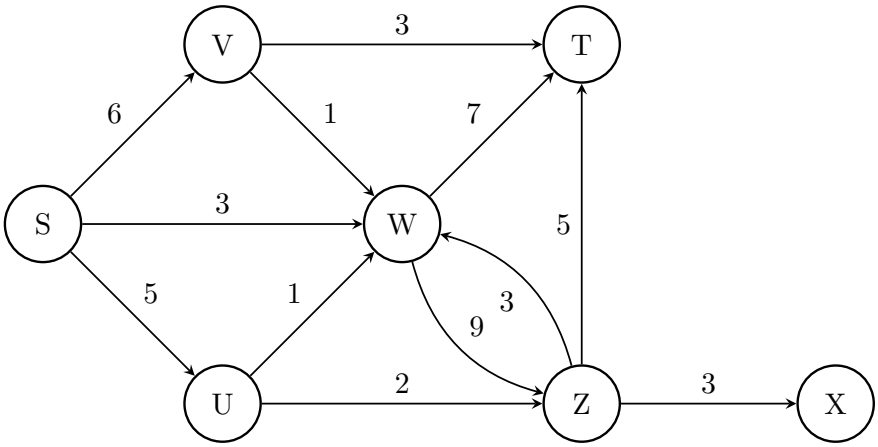
3. Un set dominante (Dominating Set) de un grafo G es un subconjunto D de vértices de G, tal que todo vértice de G pertenece a D o es adyacente a un vértice en D. El problema de decisión del set dominante implica, dado un grafo G y un número k, determinar si existe un set dominante de a lo sumo tamaño k.

   Demostrar que el Dominating Set Problem es un problema NP-Completo. Ayuda: recomendamos recordar Vertex Cover, que puede ser útil para esto.

4. Implementar un algoritmo que reciba un Grafo y un número k y devuelva un dominating set de dicho grafo de a lo sumo k vértices (si existe).

5. Hacer un seguimiento de obtener el flujo máximo en la red de transporte del dorso, realizando las modificaciones previas que fueran necesarias. Luego, definir cuáles son los dos conjuntos del corte mínimo en dicha red.


V
T
S
W
U
Z
X
3
6
1
7
3
5
5
1
3
9
2
3


V
T
S
W
U
Z
X
3
6
1
7
3
5
5
1
3
9
2
3