Temporada Ciclónica

Examen Final de Programación - Curso 2024

Día 1

La temporada ciclónica se aproxima y se prevé que habrá que suspender las clases de un día para otro. El decano está preocupado por cómo avisar a todos lo más rápido posible. En medio de un ciclón, los datos móviles podrían fallar, y es probable que algunos estudiantes necesiten que alguien camine hasta su casa para avisarles.

Para hacer esto de la manera más eficiente, el decano quiere minimizar la cantidad de avisos que se deben dar. El objetivo es asegurar que cada persona sea avisada en un máximo de k pasos a partir del decano.

Suponga que en la facultad hay N estudiantes que deben ser avisados. Si $k = 0$, esto significa que el decano debe avisar a los N estudiantes directamente, lo cuál es una opción pero evidentemente es inviable.

En caso contrario, si permitimos $k = 1$, entonces el decano puede avisar a un grupo de estudiantes de tamaño $n < N$, quienes a su vez informarán al resto de los estudiantes. Para ello, se asume que cada estudiante avisará directamente a todos sus amigos.

De la misma forma, si $k > 1$, entonces se puede avisar a un conjunto de n estudiantes, que a su vez avisarán a su primer nivel de amigos, los que a su vez avisarán a sus amigos respectivos, y así recursivamente.

Afortunadamente, se ha realizado un análisis de las interacciones en Telegram y se sabe quién es amigo de quién. Esta información está almacenada en una matriz de tipo `bool` de tamaño $N \times N$. Como es de esperar en una sociedad civilizada, la relación de amistad es simétrica, pero no necesariamente transitiva.

Dada esta información, la pregunta a responder es, para un valor k determinado de antemano $k \geq 0$, a cuántos estudiantes como mínimo debe avisar el decano para lograr que los N estudiantes reciban la información en k pasos o menos (el aviso del decano a los primeros n estudiantes no cuenta como un paso).

Descripción formal

Dada una matriz `bool[,] amigos` donde `amigos[i][j] == true` si y solo si el estudiante i es amigo del estudiante j, se busca el tamaño del conjunto más pequeño de estudiantes que deben ser avisados por el decano directamente, para asegurar que a partir de ellos todos los estudiantes se enteren en un máximo de $k \geq 0$ pasos.

Cada estudiante avisado informará a todos sus amigos. No importa si un estudiante recibe más de un aviso, solamente se cuenta el primer aviso (el que se recibe en menos pasos).

Este problema es conocido por ser NP-duro, lo que significa que muy probablemente no exista ninguna solución eficiente. Así que concéntrese en implementar un algoritmo de búsqueda exhaustiva para encontrar la solución óptima.

Implementación

Usted debe entregar un único archivo `CXX-Nombre-Apellido1-Apellido2.cs` con el siguiente contenido.

```
public static class Examen
{
    public static int MinEstudiantesAvisar(bool[,] amigos, int k)
    {
        // Implementación del algoritmo de búsqueda exhaustiva
    }

    // NOTA: Usted puede agregar en esta clase cualquier otro método auxiliar
    // que necesite, pero no puede modificar la signatura del método
    // anterior, de lo contrario el evaluador no funcionará.
}
```

Usted es libre de crear una aplicación de consola o cualquier otro tipo de proyecto que necesite para ejecutar y probar su código, pero solo debe entregar el código anterior donde esté su solución. En ningún caso entregue los casos de prueba ni código de aplicaciones que cree.

Ejemplos

Supongamos la relación de amistad representada por la matrix en Listing 1, que se puede visualizar en el grafo Figure 1, y veamos la solución óptima para diversos tamaños de K.

- Para $k = 0$ evidentemente hay que avisar a todos los estudiantes directamente, por lo que $S = 9$.
- Para $k = 1$ la solución óptima es avisar a los estudiantes 1 y 2 que a su vez avisarán al resto, por lo tanto $S = 2$.
- Para $k = 2$ lo mejor es avisar al estudiante 0, que a su vez avisa a los estudiantes 1 y 2, y estos al resto, por lo tanto $S = 1$.
- Para $k > 2$ no hay solución mejor que la anterior, por lo tanto $S = 1$.

Listing 1

```
bool[,] amigos = new bool[,]
{
    { false, true,  true,  false, false, false, false, false, false, },
    { true,  false, false, true,  false, true,  false, true,  false, },
    { true,  false, false, false, true,  false, true,  false, true,  },
    { false, true,  false, false, false, false, false, false, false, },
    { false, false, true,  false, false, false, false, false, false, },
    { false, true,  false, false, false, false, false, false, false, },
    { false, false, true,  false, false, false, false, false, false, },
    { false, true,  false, false, false, false, false, false, false, },
    { false, false, true,  false, false, false, false, false, false, },
};
```

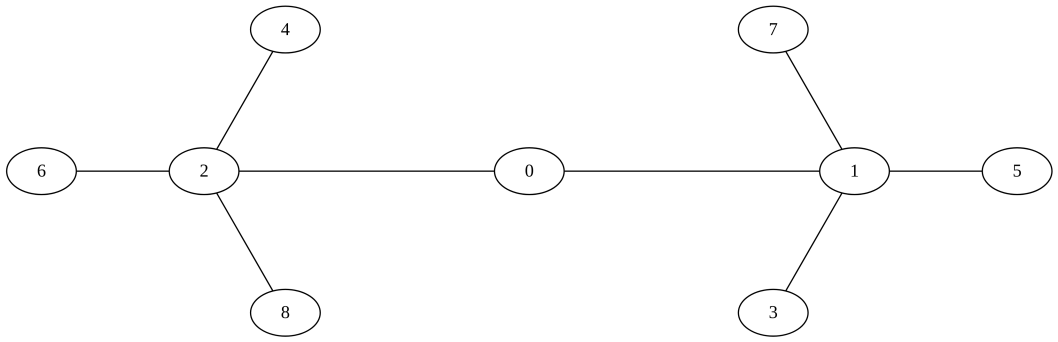

Figure 1: Grafo de ejemplo