Tarea 3: Grafos

Programación 3
Instituto de Computación
Facultad de Ingeniería de la Universidad de la República
Curso 2010
Versión 1.0

1. Objetivos

- Diseñar e implementar un tipo abstracto de datos de manera modular.
- Manejo de grafos e implementación de algunas operaciones usuales.

2. Conocimientos previos

- Metodología de programación estructurada.
- Tipos Abstractos de Datos (TAD).
- Análisis de algoritmos.
- Conocimientos básicos de C*.
- Grafos.

3. Descripción del problema

Para incentivar la creatividad e iniciativa de los estudiantes, los docentes del laboratorio de P3 hemos decido que los obligatorios resuelvan problemas reales y de importacia social. En este caso abordaremos la infelicidad en la infancia y para esto, hemos decidido ayudar a los siempre defensores de los niños: los ositos cariñosos, también conocidos como cariñositos.

La alta cúpula de Cariñolandia se reunió para resolver qué hacer ante la nueva estrategia de Sin Corazón. El malvado Sin Corazón envía a su fiel sirviente Travieso a colocar máquinas roba felicidad en diferentes puntos de una cierta región para luego controlarlas desde una máquina central.

Todas las máquinas están conectadas al menos a otra máquina y además están conectadas en forma directa o a través de otras a la máquina central, formando así una gran red roba felicidad. Los puntos

de la región donde se ubican las máquinas son los únicos puntos de la región que hay que considerar. Los cariñositos conocen los puntos de la región en donde se ubican las máquinas roba felicidad. Además conocen el tiempo de viaje entre dos pares cualesquiera de máquinas conectadas. Sólo se puede acceder a la región por los puntos de la misma que no son vigilados por Sin Corazón.

El plan de los cariñositos es el siguiente:

1. Encontrar la máquina central.
2. Encontrar el camino de máquinas que hay que seguir para llegar a la máquina central en el menor tiempo posible, comenzando en alguno de los puntos que Sin Corazón no vigila.
3. Apagar las máquinas, partiendo de la máquina central y devolviendo la felicidad a los niños.

3.1. Detalle del plan

Al presentar el plan, Corazón Luminoso (primosito mapache y el más inteligente de los cariñositos) notó al resto de los cariñositos perplejos ante tan magistral e incomprensible razonamiento. Por esto, Corazón Luminoso pasó a explicar cada paso de la estrategia, descripta aquí para facilitar el laboratorio.

3.1.1. Paso 1

Para encontrar la máquina central hay que encontrar la máquina que al ser apagada genera la mayor cantidad de redes roba felicidad. Nunca hay duda de cual es la máquina central (fig. 1).

3.1.2. Paso 2

Para encontrar el camino de máquinas que hay que seguir para ir de una máquina origen a otra destino en el menor tiempo posible es necesario aplicar el siguiente algoritmo. El algoritmo utiliza un conjunto C. En todo momento, el conjunto C contiene las máquinas que no han sido visitadas (por lo cual no se conoce el tiempo ni el camino más rápido desde el origen), y que son candidatas a ser seleccionadas en alguna etapa posterior. En un primer momento, C contiene todos los vertices menos el origen. Cuando se remueve el destino de C el problema está resuelto. En cada paso, se selecciona aquella máquina de C cuyo tiempo al origen sea mínimo (fig. 1).

El siguiente es un pseudocódigo del algoritmo.

Suponemos que:

1. las máquinas se numeran comenzando en 0,
2. hay N máquinas,
3. si no existe la arista (i, j) el costo de la misma es infinito.

Entrada:

1. o es la máquina origen
2. d es la máquina destino

Salida:

1. D - vector que almacena las distancias mínimas conocida de la máquina origen al resto
2. P - vector que almacena la máquina previa en el camino de máquinas

Inicialización:

1. $C \leftarrow \{0, 1, 2, \cdots, n\}$
2. Para $i \leftarrow 0$ hasta $N - 1$ hacer:
   a) $D[i] \leftarrow$ costo de la arista (o, i)
   b) $P[i] \leftarrow i$
3. $v \leftarrow o$

Mientras que $v \neq d$:

1. $C \leftarrow C - \{v\}$
2. Para cada $w \in C$ hacer
   a) Si $D[w] > D[v] +$ costo de la arista (v, w), entonces:
      1) $D[w] \leftarrow D[v] +$ costo de la arista (v, w)
      2) $P[w] \leftarrow v$
3. $v \leftarrow$ algún elemento de C con el menor valor en D

En el caso que hubiera más de un camino de menor costo, elegiremos aquel que comience en la máquina con menor identificador.

3.1.3. Paso 3

Para apagar las máquinas, primero se apaga la central. Luego se apagan aquellas conectadas a la central directamente comenzando por la más cercana. Si hay dos o más máquinas a la misma distancia, entonces se apaga primero las que se conectó último. El proceso se repite considerando como central a las máquinas conectadas directamente ya apagadas, hasta apagar todas las máquinas. Para devolver la felicidad a los niños se debe quedar un cariñosito en cada lugar que se apagó una máquina. Es importante conocer que cariñosito se quedó en cada máquina (fig. 2).

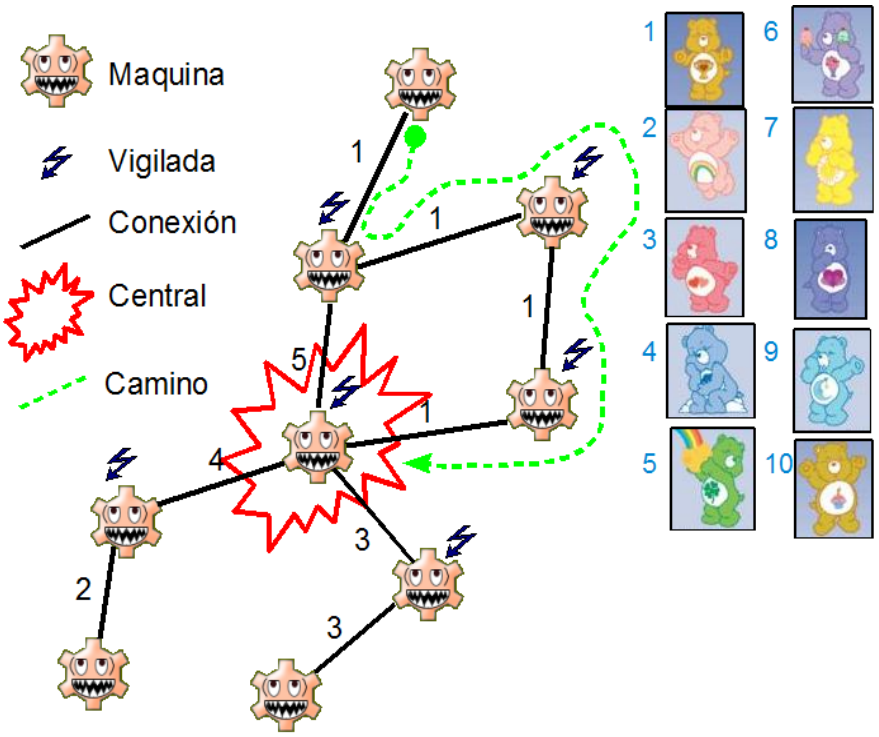

Figura 1: Red con ocho máquinas roba felicidad. Se encuentra marcada la máquina central y el camino mas corto. Sobre la derecha una lista de cariñositos.

3.2. Estructuras de datos

Se pide implementar una estructura de datos llamada **RedInfelicidad** para almacenar la información referente a las máquinas de Sin Corazón y sus respectivas conexiones. Se pide implementar además la estrucutra **ListaSimpleConPrioridad** de punteros genéricos void y algunas operaciones para utilizarla. La declaración de las estructuras se encuentran en los archivos de cabecera correspondientes.

3.3. Operaciones

Se deben implementar las operaciones incluidas en el archivo *CarinioEstrategia.h* utilizando el TAD RedInfelicidad. Estas operaciones representan los tres pasos de la estrategia a utilizar por los cariñositos.

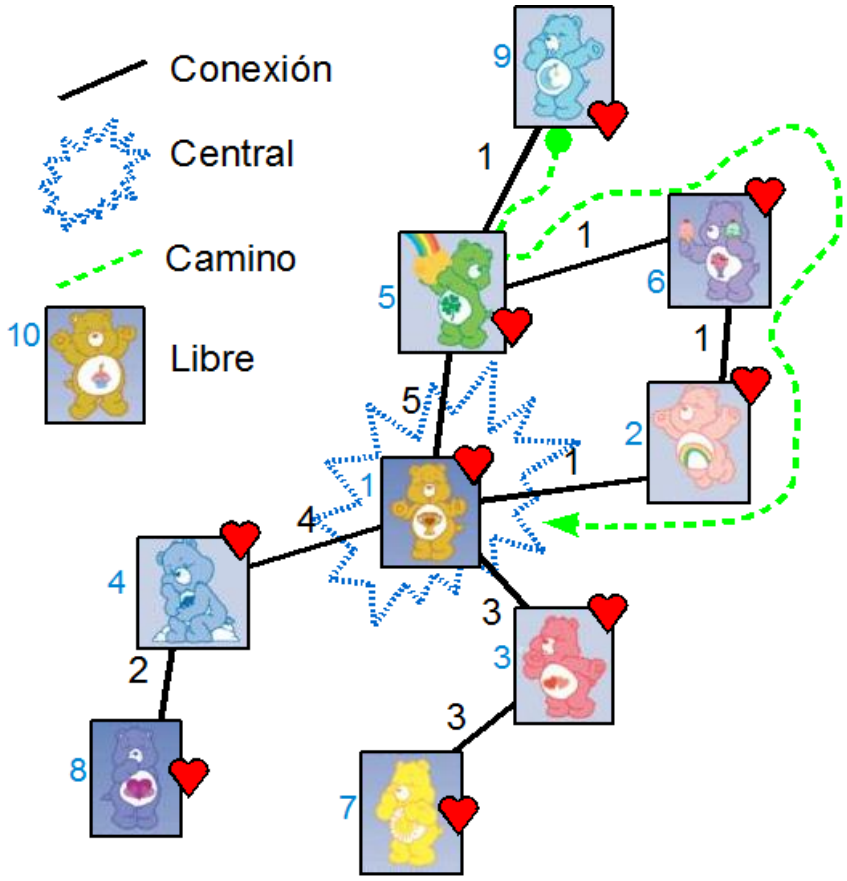

Figura 2: Resultado del tercer paso aplicado sobre la red de la figura 1.

4. Lenguaje a utilizar

El lenguaje a utilizar en este trabajo será C con las siguientes extensiones:

- Operadores new y delete.
- Pasaje de parámetros por referencia (uso de &).
- Declaración de tipos como en C++ para registros y enumerados.
- Sobrecarga de funciones.
- Uso de cin y cout.
- Uso del tipo bool predefinido en C++.

5. Qué se espera

Para cada módulo de cabecera (.h) con los prototipos de las operaciones solicitadas, debe entregarse un módulo (.cpp) con la implementación de dichas operaciones. Debe respetarse estrictamente los prototipos especificados, esto es: nombre de la operación, tipo, orden y forma de pasaje de los parámetros y tipo de retorno.

Los **módulos de cabecera pueden** bajarse de la página web del curso. Estos módulos no forman parte de la entrega, y por lo tanto, **no deben ser modificados**. **Los módulos deben funcionar en el ambiente MinGW instalado en facultad**. Se espera que todos los módulos compilen **sin errores** utilizando las flags "-Wall" , "-Werror" y "-O1", se ejecuten **sin colgarse** y den los **resultados correctos**.

6. Forma de la entrega

Se deberá entregar únicamente los siguientes archivos (respetando las mayúsculas en los nombres):

- *RedInfelicidad.cpp*
- *CarinioEstrategia.cpp*
- *ListaSimpleConPrioridad.cpp*
- *ColaSimple.cpp*

correspondiente a la implementación del sistema especificado. No se podrá entregar otros archivos que no sean estos.

La primera línea del archivo debe contener un comentario (/* ... */) con la cédula del autor, sin puntos ni dígito de verificación. Por ejemplo, si la cédula es 1.234.567-8, la primera línea de cada archivo deberá ser exactamente:

```
/* 1234567 */
```

7. Advertencia sobre el manejo de la memoria

Cuando un programa contiene errores en el manejo de la memoria, su comportamiento puede ser inestable. Esto implica que algunas veces funciona correctamente y otras no. En ciertos casos esto puede inducir a creer (erróneamente) que ciertos programas, que en realidad son incorrectos, funcionan correctamente. Este aspecto es influenciado, entre otras cosas, por el sistema operativo en el que se ejecuten los programas. Recomendamos tener sumo cuidado con este punto y testear los módulos el sistemas operativos Windows XP. CON LA VERSIÓN DE MINGW INSTALADA EN LAS SALAS DE INFORMÁTICA DE LA FACULTAD.

8. Sobre la individualidad del trabajo

El laboratorio es INDIVIDUAL. Los estudiantes pueden estudiar en grupo y consultarse mutuamente, pero NO pueden realizar en grupo las tareas de codificación, escritura, compilación y depuración del programa.

Los trabajos de laboratorio que a juicio de los docentes no sean individuales serán eliminados, con la consiguiente pérdida del curso, para todos los involucrados. Además todos los casos serán enviados a los órganos competentes de la Facultad, lo cual puede acarrear sanciones de otro carácter y gravedad para los estudiantes involucrados.

No existirán instancias en el curso para reclamos frente a la detección por parte de los docentes de trabajos de laboratorio no individuales, independientemente de las causas que pudiesen originar la no individualidad. A modo de ejemplo y sin ser exhaustivos: utilizar código realizado en cursos anteriores u otros cursos, perder el código, olvidarse del código en lugares accesibles a otros estudiantes, prestar el código o dejar que el mismo sea copiado por otros estudiantes, dejar la terminal con el usuario abierto al retirarse, enviarse código por mail, etc. Asimismo, se prohíbe el envío de código al grupo de noticias del curso, dado que el mismo será considerado como una forma de compartir código y será sancionada de la manera más severa posible.

Es decir que se considera a cada estudiante RESPONSABLE DE SU TRABAJO DE LABORATORIO Y DE QUE EL MISMO SEA INDIVIDUAL. NO CONFIAR en el borrado automático del directorio pub de las salas Windows. Es decir antes de cerrar sesión borrar todos los archivos del directorio.

9. Fecha de entrega

El trabajo debe entregarse el día **lunes 11 de octubre de 2010 antes de las 22:00 horas**. La entrega se realizará mediante un formulario que se habilitará oportunamente en la página web del curso.