

Facultad de Ciencias de las Ingenierías

Departamento de Ingeniería de Sistemas y Computación

Tarea 1 (T1):

Karger Min Cut

Por:

Ansël E. Corona Coste 2014-0031

Profesor:

J.R Felipe Nuñez

Asignatura:

Diseño y Análisis de Algoritmos

ISC-405-T-001

Santiago de los Caballeros

República Dominicana

**Descripción del problema:**

***“Tenemos el archivo kargerMinCut.txt***

*El archivo contiene la representación como lista de adyacencia de un grafo no dirigido sencillo. Hay 200 vértices etiquetados de 1 a 200. La primera columna del archivo representa la etiqueta vértice, y la fila en particular (las otras entradas, excepto la primera columna) le dice a todos los vértices de los que el vértice es adyacente a. Así, por ejemplo, la sexta fila se ve así: "6 155 56 52 120 ......". Esto sólo significa que el vértice con etiqueta 6 es vecino de (es decir, tiene arista con) los vértices con etiquetas 155,56,52,120, ......, etc.  Su tarea consiste en codificar y ejecutar el algoritmo de contracción aleatorizado para el problema de corte mínmo y utilizarlo en el grafo anterior para calcular el corte min. (SUGERENCIA: Tenga en cuenta que debes averiguar una implementación de contracción de aristas Inicialmente, es posible que desee hacer esto, ingenuamente, la creación de un nuevo grafo a partir del viejo cada vez que hay una contracción de aristas Pero también se debe pensar más en implementaciones más eficiente.. i) (ADVERTENCIA:.. favor asegúrese de ejecutar el algoritmo muchas veces con diferentes semillas aleatorias, y recordar el corte más pequeño que alguna vez se encontró). De una respuesta de este valor encontrado”*

En síntesis, se nos pide desarrollar una implementación del algoritmo de contracción aleatoria aplicada al problema de corte minimo.

Utilizando el algoritmo de Karger (que es un algoritmo probabilístico), producimos una solución correcta en la *mayoría* de los casos, en este reporte veremos varias corridas.

**Desarrollo:**

Tenemos un archivo el cual contiene la representación como lista de adyacencia de un grafo no dirigido sencillo. Hay 200 vértices etiquetados de 1 a 200. La primera columna del archivo representa la etiqueta vértice, y la fila en particular (las otras entradas, excepto la primera columna) le dice a todos los vértices de los que el vértice es adyacente a. Así, por ejemplo, la sexta fila se ve así: "6 155 56 52 120 ......". Esto sólo significa que el vértice con etiqueta 6 es vecino de (es decir, tiene arista con) los vértices con etiquetas 155,56,52,120, ......, etc…

El algoritmo se basa en un procedimiento de contracción borde. Este procedimiento es, dado un grafo G y una e = borde de fusionar a y b para formar un solo tapa ab cuyos vecinos son la concatenación de los vecinos de a y b. El gráfico formado se denota G / e. Esta transformación puede revelar múltiples bordes, que está trabajando con dispositivos multiplicadores. El punto importante es que un corte en el gráfico después de la contracción ya era un corte en el gráfico original.

El algoritmo es simplemente contratar a un borde dibujado de manera uniforme en el gráfico con el contrato, y repetir el proceso hasta que sólo tienen dos picos. Los dos picos representan la partición de vértices y los bordes entre ellos se encuentran los bordes de corte.

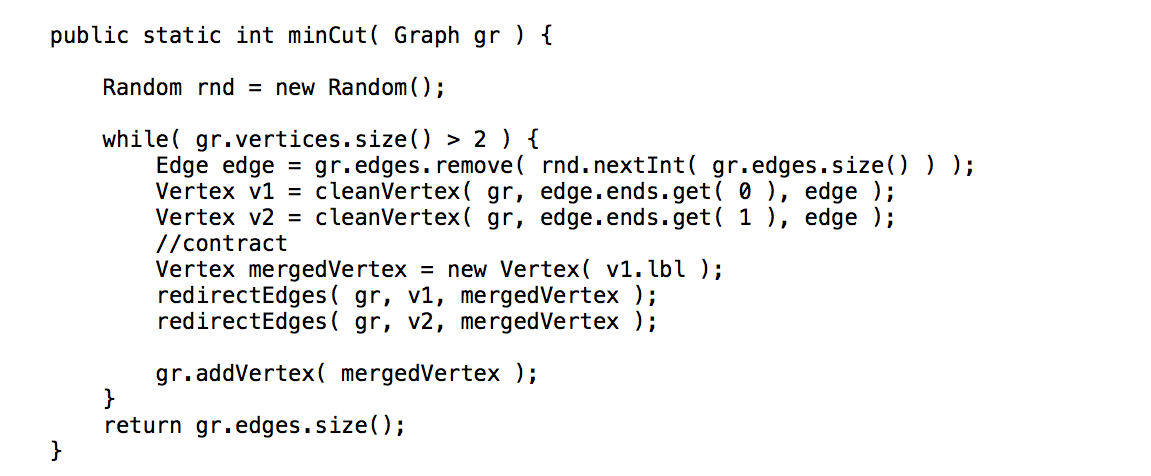
La solución de este problema viene de la siguiente manera, tenemos una entrada, en la cual tenemos una representación en listas de adyacencia de un grafo no dirigido sencillo, tenemos 200 vértices etiquetados de 1 a 200. La primera columna del archivo representa el vertice y los numeros que siguen en cada fila representan los vertices adyacentes, ejemplo: "6 155 56 52 120 ......", Esto sólo significa que el vértice con etiqueta 6 es vecino de (es decir, tiene arista con) los vértices con etiquetas 155,56,52,120

El algoritmo esta basado sobre un procedimiento de contracción borde que funciona de la siguiente forma, dado un grafo G y una e = borde de fusionar a y b para formar una sola tapa “ab” cuyos vecinos serán la unión de los vecinos de a y b. El grafo resultante será denominado G / e, durante esta transformación se revelan multiples bordes que a su vez trabajan con dispositivos multiplicadores, importante recalcar que un corte en el grafo despues de la contracción ya era un corte en el grafo original.

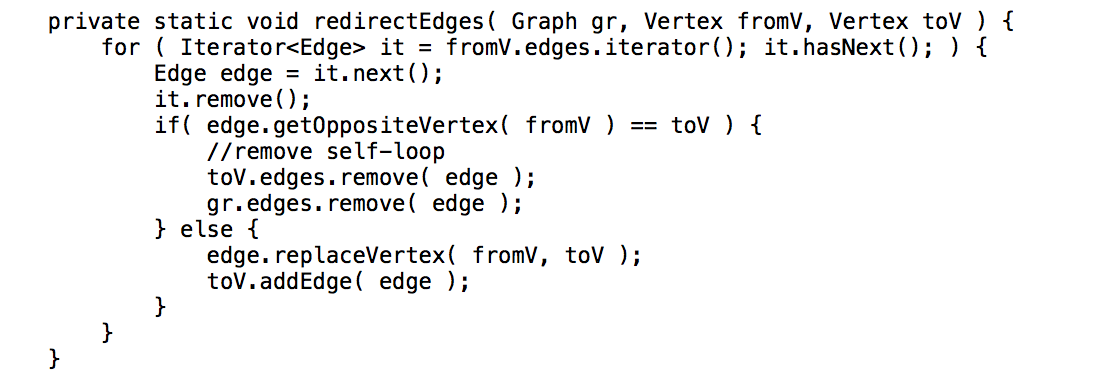
Este algoritmo basicamente de manera iterativa contrae aristas elegidas aleatoriamente hasta que solo quedan dos nodos, los cuales representan un corte en el grafo original. Iterando este algoritmo una cantidad especifica de veces podemos encontrar con una probabilidad alta el numero minimo de cortes.

Para la implementación y codificación del problema elegí el lenguaje Java.

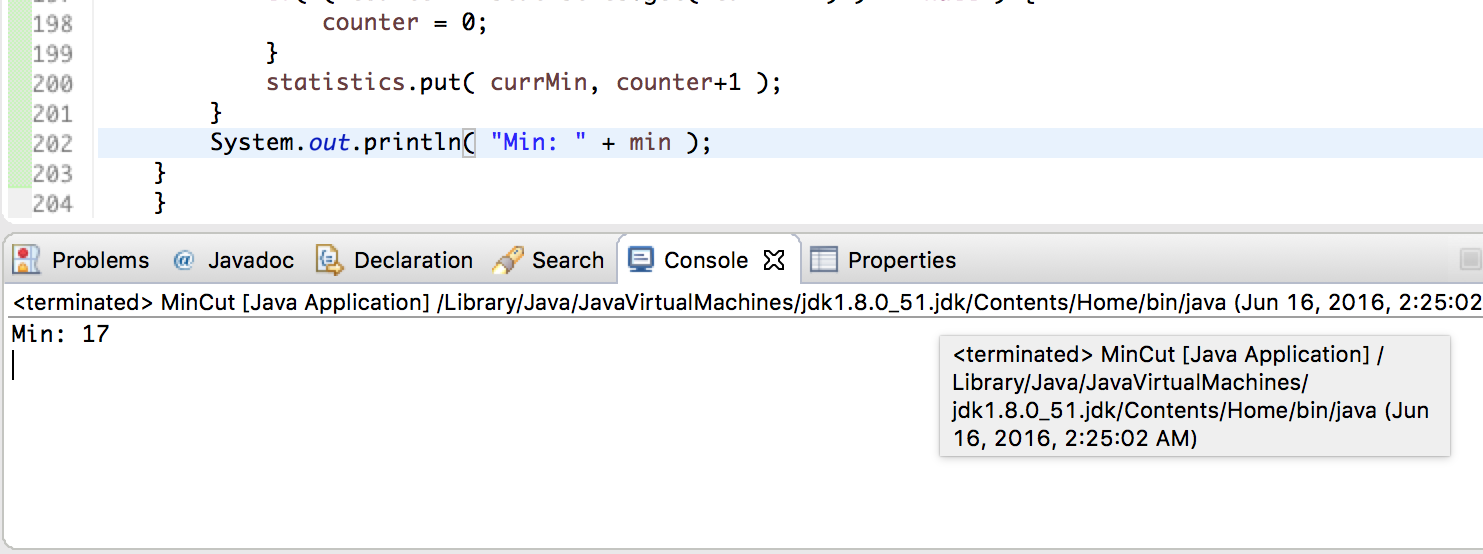
Aquí una captura del algoritmo codificado:



Una función que ayuda en la contracción de aristas:



Resultado de la corrida:



A mi laptop le tomó **8 minutos** correr el programa, por lo que lo corrí solo dos veces y me dio 17 las dos veces.