Lucia Pardo - 99999
Federico del Mazo – 100029
23 de junio del 2017

Facultad de Ingeniería de la Universidad de Buenos Aires

Año 2017 - 1er Cuatrimestre

Algoritmos y programación II - 75.41

Cátedra Wachenchauzer

Trabajo Práctico N.° 3

El trabajo consiste en dos partes. Por un lado, el TDA Grafo, por el otro, un programa que haciendo uso de un grafo modelo de la red social Youtube, aplica una serie de funciones sobre este, devolviendo estadísticas, recomendaciones y demás

TDA Grafo:

La clase grafo esta pensada como un diccionario de listas, siendo los vértices las claves y sus adyacencias los valores. Este TDA tiene los siguientes métodos principales:

- Agregar vértice: Simplemente agrega un indice al diccionario
- Borrar vértice: Recorre todo el diccionario borrando el vértice de las distintas listas de adyacencias y finalmente borra el vértice del diccionario
- Agregar arista: Siendo un grafo no dirigido, agrega el primer vértice la lista del segundo y viceversa. Si los vértices no existen, los agrega
- Borrar arista: Va a ambas listas de adyacencias y borra el vértice respectivo
- Random walk: Recibe un vértice de origen y una cantidad de pasos y, al azar, hace los pasos necesarios desde el origen. Devuelve una lista del camino que hizo
- BFS: Devuelve un diccionario de una búsqueda en anchura desde el origen recibido como parámetro
- Label propagation: Para identificar grupos de vértices altamente conectados, le indica una etiqueta a cada vértice y luego va propagandolaa por todo el resto del grafo, haciendo que los vértice altamente conectados terminen con la misma etiqueta indicadora
- len: La longitud de la clase es la lista de vértices

- Iter: “For v in grafo” recorre cada vértice del grafo
- Adyacentes: Devuelve la lista de adyacentes del vértice

Además de todo lo listado tiene métodos mas simples como “cant_aristas” o “pertenece” que solo hacen un llamado y devuelven un booleano o una cantidad y un iterador (que solo recorre el diccionario de vértices)

Funciones del programa:

Partiendo de un main que recibe un archivo por comando, se genera un grafo acorde al archivo recibido y se espera por input (sea teclado o texto de instrucciones) los comandos a hacer. Cada comando posible esta en un diccionario de comandos y cada vez que es llamado la función respectiva se ejecuta. Los comandos son:

- Funciones random walk: Siendo funciones que tienen un comportamiento muy parecido, esta función es llamada por un wrapper que indica que función es la utilizada y simplemente cambia una linea de código, según la función a usar. Las funciones son:
  - Similares: Hace una cantidad (definida por constante) de random walks y devuelve una lista de los usuarios mas similares al dado
  - Recomendar: Comprueba que de esta lista los usuarios no tengan conexión al original
  - Centralidad: Los random walks ahora son al azar, con la idea de que los usuarios a los que mas se llega de manera aleatoria son los mas influyentes.
- Comunidades: Haciendo uso del label propagation, determina el grupo de vértice mas conexos del grafo (marcadas por constante las cotas) y devuelve una lista de las comunidades formadas
- Camino: Devuelve uno de los posibles caminos mas cortos entre la partida y la llegada, haciendo uso de un BFS sobre el grafo
- Distancia: Devuelve un diccionario de los vértices y sus distancias desde el origen
- Estadísticas: Imprime las estadísticas del grafo, siendo estas la cantidad de vértices, la cantidad de aristas, el grado promedio de salida de cada vértice, y la densidad del grafo.