Tarea No. 4. Procesamiento de Grafos

LUIS DANIEL ROA GONZÁLEZ, Instituto Tecnológico de Estudios Superiores de Monterrey, Campus Santa Fe

El motivo de esta tarea es, poder identificar que tipo de algoritmo para convertir un documento de texto a diferentes tipos de grafos es el mas eficiente, es decir, cual es el mas rápido, el mas eficiente y tanto las ventajas como las desventajas de cada uno de los formatos utilizados.

Como meta adicional, se aprendió a utilizar el formato oficial de la ACM (Association for Computing Machinery) y se llevó a cabo utilizando LaTeX .

Additional Key Words and Phrases: grafos, tarea, analisis, diseño, algoritmos, ITESM, LaTeX

ACM Reference Format:
Luis Daniel Roa González. 2019. Tarea No. 4. Procesamiento de Grafos. 1, 1 (October 2019), 3 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCCIÓN

Esta tarea tuvo como finalidad, poder convertir un *dataset* que escogieramos de la página de Snap, la cual ofrece diferentes tipos de *datasets* para que se pueda conllevar una simulación acerca del funcionamiento de la red elegida.

Para el propósito de esta investigación, se llevó a cabo usando el dataset llamado **wiki-Vote**, el cual consiste en la cantidad de votos que se llevaron a cabo para poder aceptar a miembros nuevos como administradores para que puedan editar el sitio a su gusto. Este dataset contiene información desde el momento en el cual Wikipedia surgió (2001) hasta la fecha en la que se recolectó esta información (2008).

Al llevar a cabo esta tarea, se descargó un programa para poder abrir documentos con formatos de grafos, este programa se llama **Gephi**, este mismo permite visualizar el sistema de grafos.

Para conllevar de manera correcta esta tarea, fue requerido realizar un programa en C++ que pudiera convertir el documento de texto con la información al respecto del grafo, a los siguientes formatos:

- GraphML
- GEXF
- GDF
- JSON

2 PROGRAMA REALIZADO

Como se mencionó en la introducción, se realizó un programa donde se inserta el nombre del documento de texto (*con terminación .txt*) el cual contiene el nodo principal y el nodo al que se encuentra conectado. El programa está compuesto usando dos diferentes archivos, un *main* con

Author's address: Luis Daniel Roa González, a01021960@itesm.mx, Instituto Tecnológico de Estudios Superiores de Monterrey, Campus Santa Fe, Av Carlos Lazo 100, Santa Fe, La Loma, Santa Fe, Ciudad de México, 01389.

XXXX-XXXX/2019/10-ART $15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

terminación **.cpp**, este lleva el nombre de *Tarea4.cpp*, la única funcionalidad de este programa es medir los tiempos y mandar a llamar los métodos que se están utilizando para la creación de los documentos que contendrán los grafos.

El otro programa del que se menciona, es un programa tipo *header*, en este se encuentran los métodos donde se convierte el documento de texto a los formatos que se especificaron en la introducción de este documento. Cada método corre después de que el anterior finalizó.

2.1 Conversión de *.txt* a *GrapML*

La manera en la que se tiene programado el método para convertir el documento de texto en un documento de tipo GraphML resultó ser la mas eficiente en el sentido que el documento se pudo abrir exitosamente en Gephi, el visualizador que se solicitó usar para poder realizar esta tarea.

La complejidad de este algoritmo resultó ser de $O = |O| + |V|$.

2.2 Conversión de *.txt* a *GEXF*

El método para convertir el documento de texto al formato GEXF duró unos segundos menos que el del formato anterior, pero hubo un inconveniente al correrlo en la aplicación de Gephi, el problema principal fue que se tuvo que modificar la cantidad de memoria que estaba siendo usada por la aplicación para que se pudiera desplegar la información de manera correcta.

La complejidad de este algoritmo resultó ser de $O = |O| + |V|$.

2.3 Conversión de *.txt* a *GDF*

A los métodos anteriores se les reconoció por ser mas veloces que el anterior, pero este fue el más veloz de los algoritmos de los que se habló hasta el momento. Este mismo si se pudo abrir y visualizar en *Gephi*, pero fue el que mas se tardó en poder ser visualizado.

La complejidad de este algoritmo resultó ser de $O = |O| + |V|$.

2.4 Conversión de *.txt* a *JSON*

Por último, el algoritmo para convertir el documento de texto a un formato JSON igual tuvo bastante rapidez, puesto que se tardó solamente **26 ms** en terminar de correr. Desafortunadamente, este formato no pudo ser visualizado en Gephi debido a que el formato .JSON no es aceptado en el visualizador.

La complejidad de este algoritmo, al igual que el de los anteriores, resultó ser de $O = |O| + |V|$.

2.5 Tabla de comparación de tiempos

GraphML	GEXF	GDF	JSON
34 ms	30 ms	15 ms	26 ms

Tiempo medido en milisegundos

3 GEPHI Y SUS FUNCIONALIDADES

El visualizador de nodos, Gephi, fue muy útil para esta tarea, debido a que no es muy complicado de usar y las herramientas internas son sencillas de entender, pero si se encontraron dificultades. Como alguien que nunca ha usado este tipo de visualizadores, especificamente Gephi, puedo decir que estoy agradecido que existan semejantes programas.

Dentro de Gephi se pueden encontrar diferentes opciones para poder visualizar el documento, si este es compatible. Se puede ver que nodo esta conectado a cual, como se puede visualizar toda la población, etc...

Aunque se hablen cosas buenas de este, desgraciadamente tambien tiene sus inconsistencias, por ejemplo, no se pudo abrir/correr el documento JSON. Al igual que este último detalle, al momento de intentar correr un documento mas grande, se debe modificar la memoria que este usa, pero no hay instructivo donde se mencione como se debe modificar de manera correcta.

4 REFERENCIAS

4.1 Código

El código fue obtenido de:

https://github.com/margotduek/AnalisisDisenioDeAlgoritmos/tree/master/tarea4

Este mismo fue modificado para que funcionara a base de las especificaciones.

4.2 Gephi

Gephi fue obtenido de:

https://gephi.org/

4.3 LaTeX

La distribución de LaTeX no fue descargada a la computadora, por lo tanto se uso la distribución encontrada en línea llamada:

https://www.overleaf.com.