Universidad de San Carlos de Guatemala
Escuela de Ingeniería en Ciencias y Sistemas, Facultad de Ingeniería
Introducción a la programación y computación 2, 1er. Semestre 2021.

IMPLEMENTACIÓN DE TIPOS DE DATOS ABSTRACTOS PARA ESTABECER UN MÉTODO QUE FACILITE LA TRANSMISIÓN DE DATOS

201908686 – Gerson Sebastian Quintana Berganza

Resumen

El costo total en la transmisión de datos para el procesamiento de aplicaciones es un factor muy importante por lo que establecer un método que pueda minimizar esta transmisión es necesario; y esta es la esencia del proyecto, aplicar una metodología de agrupamiento que permita un mejor procesamiento de los datos. Para realizar lo anterior se desarrolló una aplicación que capaz de generar una estructura y a partir de esta, obtener matrices haciendo la analogía de que las filas corresponden a tuplas y dentro de estas, hay *n* cantidad de sitios que, como se mencionó anteriormente representan con la frecuencia se ha accedido a estos sitios. Para su implementación se utilizaron tipos de datos abstractos (TDA's) para almacenar cada uno los datos y generar una estructura de matriz. Realizando ciertas operaciones de comparación se obtiene una matriz reducida correspondiente a cada una de las matrices. Tantos los archivos de entrada como los archivos de salida se pueden visualizar tanto en gráficas como en un nuevo archivo.

Palabras clave

1. TDA (Tipo de Dato Abstracto)
2. XML (Lenguaje de Etiquetado Extensible) Agrupamiento

Abstract

The total cost of data transmission for application processing is a very important factor, so establishing a method that can minimize this transmission is necessary; And this is the essence of the project, to apply a grouping methodology that allows better data processing. To do the above, an application was developed that is capable of generating a structure and from this, obtaining matrices making the analogy that the rows correspond to tuples and within these, there are n number of sites that, as mentioned above, represent with the these sites have been accessed frequently. For it is implementation, abstract data types (TDA's) were used to store each data and generate a matrix structure. Carrying out certain comparison operations, a reduced matrix corresponding to each of the matrices is obtained. Both the input files and the output files can be displayed both in graphs and in a new file.

Keywords

1. ADT (Abstract Data Type)

2. XML (Extensible Markup Language) Grouping.

Introducción

En la actualidad, la optimización de los recursos utilizados en las empresas no solo en cuestiones dirigidas a, por ejemplo, el equipo de cómputo, servidores, etc. Sino que también en la transmisión de datos ya que esto trae consigo muchos beneficios, como el requerimiento de tiempo y la fuerte demanda de memoria.

En este caso, en el contexto en que existen diferentes sitios en los cuales se realizan muchas consultas, enviar las consultas realizadas a estos sitios sin ningún proceso intermedio que permita estructurarlos de una manera eficiente generaría muchas desventajas ya que los datos enviados serían muchos y las aplicaciones dedicadas a analizarlos tardarían más en procesarlos. Y este es básicamente el objetivo del proyecto: generar un método que permita obtener, a partir de un conjunto de datos que cumplan un determinado patrón una especie de resumen que facilite el análisis de estos datos.

Desarrollo del tema

La implementación de un método que permita optimizar la transmisión de datos, requiere de seguir un conjunto de reglas y patrones que rijan las operaciones a realizar en los datos. Estas operaciones pueden ser generales o más específicas, como las que dependen de cierta estructura. Esta estructura para fines de este proyecto, solo puede ser un tipo de dato abstracto. “Un Tipo Abstracto de Datos (TAD) corresponde a una clase de objetos establecida en el proceso de abstracción de datos y viene dado por su especificación y una implementación que la satisfaga” (Arraiz, 2002, p.23).

En primer lugar, para implementar un TDA primero hay que establecer los tipos de datos a almacenar. Debido a que la idea gira en torno a agrupar la frecuencia de acceso a determinados sitios, los valores almacenados son de tipo entero en donde toda la información requerida se obtiene a partir de un archivo de entrada con extensión xml con una estructura predefinida. La estructura de este archivo define con claridad en donde empieza y termina una matriz de acceso.

La solución al problema que se implementa en la aplicación y consiste en tres principales procesos:

1. Procesamiento del archivo:
Este es el primer proceso que se ejecuta y es el más importante, ya que es donde se analiza el archivo de entrada validando que ningún nombre de matriz se repita y que los datos ingresados sean congruentes a las dimensiones de la matriz especificados en atributos de esta.

Para validar que ninguna matriz dentro del documento se repita se implementó una lista circular simplemente enlazada. “Una lista enlazada simplemente es en la que el último elemento (cola) se enlaza al primer elemento (cabeza) de tal modo que la lista puede ser recorrida de modo circular (anillo)” (Manuel, p.31).

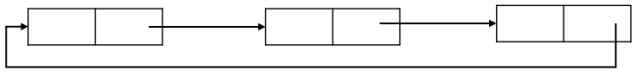

Figura 1. Lista circular simplemente enlazada.

Fuente: elaboración propia, 2020.

Para ello, se guardaron todos los nombres de las matrices en el archivo independientemente de si estaba repetido el nombre o no. Una vez guardados todos los nombres únicamente se validó que cada uno

de ese nombre existiera una sola vez, se lo contrario se le informa al usuario.

Además, es en este proceso en donde se implementa las librerías *ElementTree* y *minidom*, la cuales facilitan la lectura del documento xml. "Un documento XML es una estructura jerárquica autodescriptiva de información, que consiste en un conjunto de átomos, elementos compuestos y atributos (Dalamagas et al., 2006)", no solo para validar lo anterior, sino que también para enviar cada uno de los datos que se encuentran dentro de la etiqueta *dato* y asignándoles una posición dentro de la matriz. No solamente se envían los datos así, sino que también validando que sean enteros positivos y que la cantidad de datos sea congruente con las dimensiones de la matriz, es decir, que no haya ni menos ni más datos que los permitidos dentro de la matriz.

Para la generación de la matriz, se utilizaron dos listas enlazadas: una para los datos originales que contiene el archivo y otra para la matriz binaria. "Una lista simplemente enlazada cada nodo conoce al nodo siguiente, de forma tal que es unidireccional su recorrido" (Méndez, p.8). Dentro de la matriz binaria se almacena, tal y como su nombre lo indica, únicamente un 1 cuando el dato de entrada en esa posición sea diferente de cero o un 0 cuando el dato de entrada en esa posición sea igual a 0.

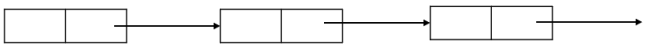

Figura 2. Lista simplemente enlazada.

Fuente: elaboración propia, 2020.

La forma en que actúan estas dos matrices es de la siguiente forma: se obtiene, cada una de las filas de la matriz binaria y esta se compara con todas las demás, si cada uno de los valores en posiciones de esa fila es igual cada uno de los valores en las posiciones de otras filas, estas filas, pero esta vez en la matriz con los datos originales se suman en sus respectivas posiciones. Y así sucesivamente con cada una de las filas para generar una matriz reducida que representará en cierta forma un resumen de qué filas cumplen con un patrón en específico.

Esta matriz reducida no se guarda en la misma lista circular, en cambio, se implementa una nueva en la que se almacena el nombre de cada matriz y en la que cada matriz tiene asociada una lista simplemente enlazada que se utiliza al igual que la anterior: como una matriz.

2. Escritura de un archivo de salida:
Este proceso es dependiente del anterior ya que sin que los datos no se hayan procesado aún, el archivo de entrada no puede generarse.

Este proceso, en resumen, lo que realiza es tomar la lista circular generada en el proceso de análisis del archivo y a partir de cada uno de los nombres matrices obtiene cada uno de los elementos de la lista enlazada (que se utilizó como matriz) en cada una de las posiciones y con ayuda nuevamente de las librerías *ElementTree* y *minidom* se escribe nuevamente, (siempre con la estructura del archivo de entrada) las matrices reducidas obtenidas a partir de determinados patrones.

3. Generación de gráfica:
La lógica de este proceso es un poco distinta a la de los procesos anteriores ya que, aunque valida por medio de la matriz circular que no haya ninguna matriz con el mismo nombre, obtiene la posición de una matriz dentro del archivo de entrada y vuelve a leer la matriz nuevamente.

Los datos leídos nuevamente no son almacenados en ningún tipo de datos abstracto, en vez de eso se van concatenando a una variable que por medio de ciertas manipulaciones crea una estructura que permite generar por medio de la librería *Graphviz* un grafo dirigido correspondiente a la matriz elegida. “Un Grafo Dirigido (o digrafo) G es un par (V, E), donde V es un conjunto finito y E es una relación binaria sobre V. Es decir, E es un subconjunto del producto cartesiano VxV” (González, 2002, p.2).

Conclusiones

1. El crear métodos que faciliten no solo envío de datos, sino que también organizados es una tarea fundamental para procesos que requieren fuertes transmisiones de estos; por eso, saber implementar estos métodos es tan importante ya que optimizan su tiempo y procesamiento.

2. La utilización de estructuras generadas sin la utilización de herramientas que distintos lenguajes ponen a disposición es parte esencial ya que no solo permite conocer el funcionamiento interno de estas estructuras, sino que, dependiendo de la estructura generada, realizar diferentes operaciones con los datos que permitan optimizar los procesos.

3. La implementación de métodos aplicados en datos que hagan la analogía de casos que se pueden presentar el futuro son una buena forma de introducción hacia el manejo de información y como esta puede ser manipulada para que estos puedan ser analizados y procesados un el menor tiempo posible.

Referencias bibliográficas

Arraiz, E., Pasarella, E., & Zoltan, C. (2002). Tipos Abstractos de Datos y Algoritmos. Universidad Simón Bolívar, Sartenejas, Venezuela: http://www. ldc. usb. ve/meza/ci, 2616.

DALAMAGAS, T., CHENG, T., WINKEL, K.-J. & SELLIS, T. 2006. A Methodology for Clustering XML Documents by Structure. Information Systems.

González, A. (2002). Definiciones: conjuntos, grafos, y árboles. Recuperado el, 29.

Manuel, L. M. V. Unidad II. Estructuras de Datos Lineales-Listas.

Méndez, M. 75.41 Algoritmos y Programación II Tda Lista y sus Derivados.