|  |
| --- |
| **PROYECTO NO. 1** |
| **201901429 – Diego Abraham Robles Meza** |

**Resumen**

El problema de diseño de distribución consiste en determinar el alojamiento de datos de forma que los costos de acceso y comunicación son minimizados. Como muchos otros problemas reales, es un problema combinatorio NP-Hard. Algunas de las situaciones comunes que hemos observado cuando se resuelven instancias muy grandes de un problema NP-Hard son: Fuerte requerimiento de tiempo y fuerte demanda de recursos de memoria. Un método propuesto para resolver este tipo de problemas consiste en aplicar una metodología de agrupamiento.

Para “nt” tuplas y “ns” sitios, el método consiste en tener la matriz de frecuencia de acceso en los sitios F[nt][ns] de la instancia objetivo, transformarla en una matriz de patrones de acceso y agrupar las tuplas con el mismo patrón.

El patrón de acceso para una tupla es el vector binario indicando desde cual sitio la tupla es accedida.

**Palabras clave**

NP-Hard

TDA

Matriz

xml

***Abstract***

The distribution design problem consists of determining the data hosting such that access and communication costs are minimized. Like many other real problems, it is a combinatorial NP-Hard problem. Some of the common situations we have observed when solving very large instances of an NP-Hard problem are: Strong time requirement and strong demand on memory resources. A proposed method to solve this type of problems consists in applying a clustering methodology.

For "nt" tuples and "ns" sites, the method consists of having the access frequency matrix at the sites F[nt][ns] of the target instance, transforming it into an access pattern matrix and clustering the tuples with the same pattern.

The access pattern for a tuple is the binary vector indicating from which site the tuple is accessed.

***Keywords***

Np-Hard

TDA

Matrix

xml

**Introducción**

Uno de los problemas que se presentan a la hora de trabajar con sitios basados en bases de datos son los problemas NP-Hard que pertenecen a la clase NP. La clase NP son problemas de complejidad en la toma de decisiones, búsqueda y optimización de aplicaciones. Debido a esto han surgido varias maneras de dar solución a estos problemas, y de los cuales estaremos utilizando uno de ellos. El utilizado en este proyecto se basa en el calculo de la matriz de frecuencia de acceso para así optimizar la ruta requerida y minimizar los gastos de memoria del equipo. Asimismo, se utiliza la programación orientada a objetos para crear listas TDA.

**Desarrollo del tema**

La clase de complejidad NP es el conjunto de problemas que pueden ser resueltos en tiempo polinómico por una maquina de Turing no determinista. La importancia de esta clase de problemas de decisión es que contiene muchos problemas de búsqueda y de optimización para los que se desea saber si existe una cierta solución o si existe una mejor solución que las ya conocidas.

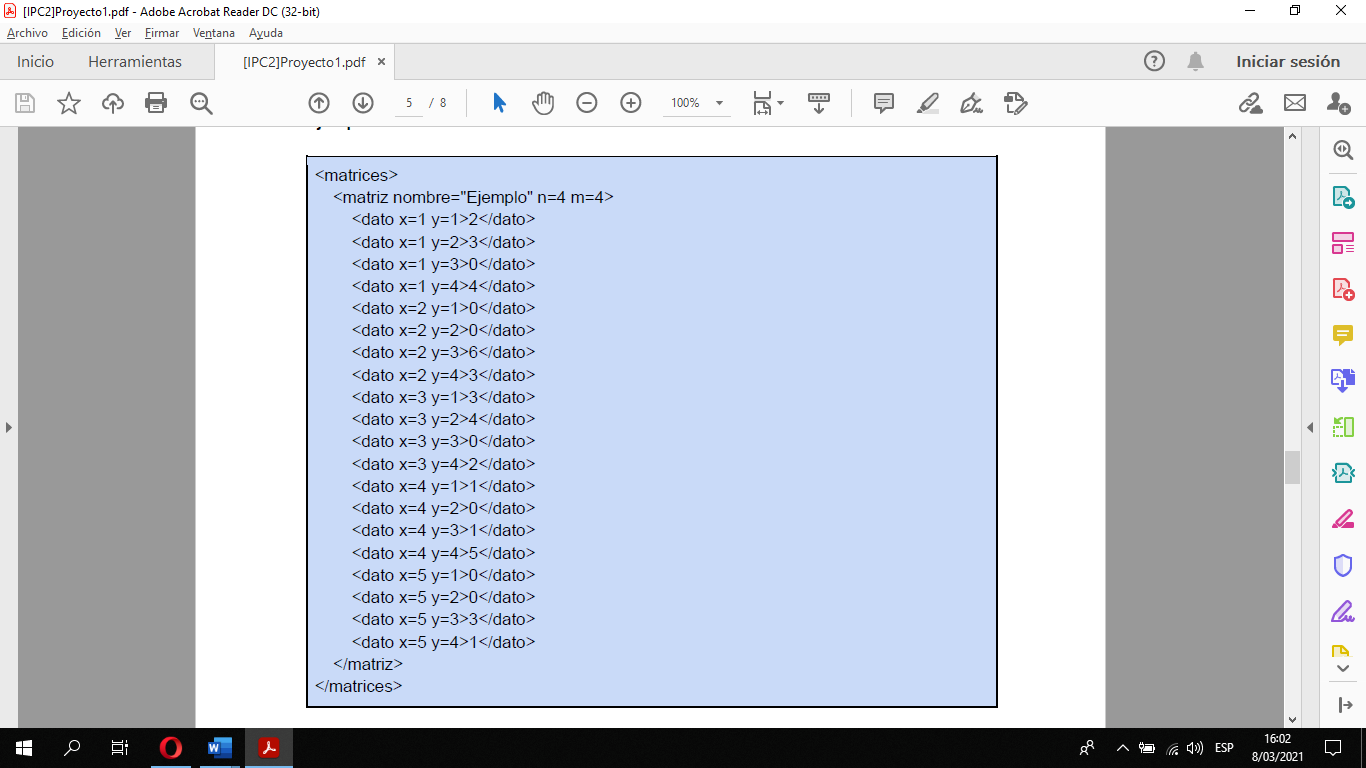
La importancia de esta clase de problemas ha hecho que se creen algoritmos para decidir en tiempo polinómico. Sin embargo, debido a su amplia extensión hay algunos problemas que no se pueden solucionar de una manera más eficaz que empleando los algoritmos ya conocidos.

**Xml:**

Xml es el acrónimo de Extensible Markup Language, es decir, es un metalenguaje que permite definir lenguajes de marcas desarrollado por el World Wide Web Consortium (W3C) utilizado para almacenar datos en forma legible. A diferencia de otros lenguajes, XML da soporte a bases de datos, siendo útil cuando varias aplicaciones deben comunicarse entre sí o integrar información.

Este lenguaje maneja etiquetas con atributos, la estructura va de la siguiente manera:

Se define una etiqueta padre, la cual contendrá a todas las demás etiquetas. Luego se definen las etiquetas hijas, las cuales contendrán todos los datos de las matrices. Y de ultimo van las etiquetas nodo, las cuales contendrán cada uno de los elementos de las etiquetas matrices.



*Figura 1.* Estructura xml

Fuente: Laboratorio IPC2, 2021.

**ElementTree**:

Es una librería simple para procesar archivos XML que se puede utilizar en Python. Al utilizar Element Tree, se crea una estructura en forma de árbol utilizando el comando “parse” y se obtiene un elemento raíz. Una vez identificado este elemento, ya podremos recorrer el árbol debido a que todos los nodos del árbol están conectados.

**TDA:**

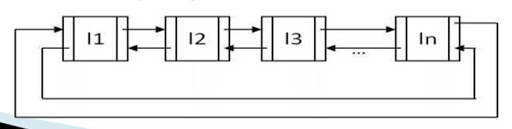
La abstracción de datos consiste en ocultar las características de un objeto y obviarlas, de manera que solamente utilizamos el nombre del objeto en nuestro programa. Esto es similar a una

situación de la vida cotidiana. Cuando yo digo la palabra “perro”, usted no necesita que yo le diga lo que hace el perro. Usted ya sabe la forma que tiene un perro y también sabe que los perros ladran. De manera que yo abstraigo todas las características de todos los perros en un solo término, al cual llamó “perro”. A esto se le llama ‘abstracción’ y es un concepto muy útil en la programación, ya que un usuario no necesita mencionar todas las características y funciones de un objeto cada vez que este se utiliza, sino que son declaradas por separado en el programa y simplemente se utiliza el término abstracto (“perro”) para mencionarlo.

Con este concepto en mente ya podemos definir que es un tipo de dato abstracto. Es un modelo matemático compuesto por una colección de operaciones definidas sobre un conjunto de datos.

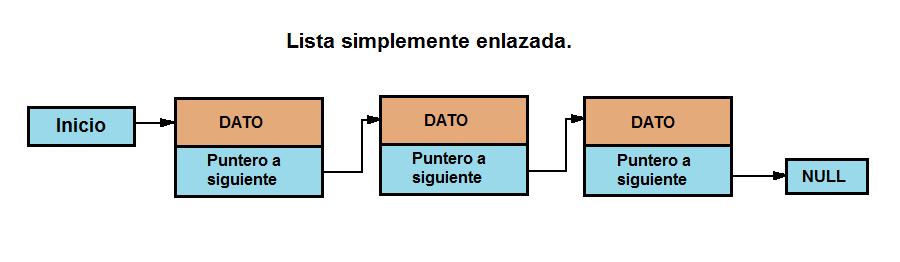
Con los tipos de dato abstracto se pueden crear listas empleando una clase llamada generalmente “Nodo”, de la cual se abstraen sus características principales. Los tipos de listas que se pueden crear son la Lista Simple, Lista Doble, Lista Circular, Lista doblemente enlazada.

En el desarrollo de el proyecto empleamos tanto las listas simples como las listas doblemente enlazadas. En las listas simples se van ingresando las matrices leídas de los archivos con extensión xml ingresados y la lista doblemente enlazada se utiliza para guardar todas las matrices previamente ingresadas en las listas simples.



*Figura 1I.* Lista Doblemente enlazada.

Fuente: Google Imágenes.



*Figura 1II.* Lista Simplemente enlazada.

Fuente: Google Imágenes.

**Matrices de acceso**:

Luego de haber almacenado correctamente todas las listas se proceden a calcular las matrices de acceso, la cual consiste en que en cada elemento de la matriz original en el que exista un elemento diferente de cero, se reemplazara por un número uno. Este calculo se logra empleando las operaciones definidas anteriormente mencionadas.

A continuación, se muestra un ejemplo de una matriz ingresada y posteriormente su matriz de acceso



*Figura 1V.* Matriz original.

Fuente: Laboratorio IPC2, 2021.



*Figura V.* Matriz de acceso.

Fuente: Laboratorio IPC2, 2021.

**Matriz reducida:**

La matriz reducida consiste en que con los datos de la matriz de acceso se obtiene una fila, y esta fila se va comparando con todas las demás. Cuando varias filas son iguales, se procede a sumar los valores de las filas que son iguales en la matriz original. Dando resultado a una matriz como la siguiente:

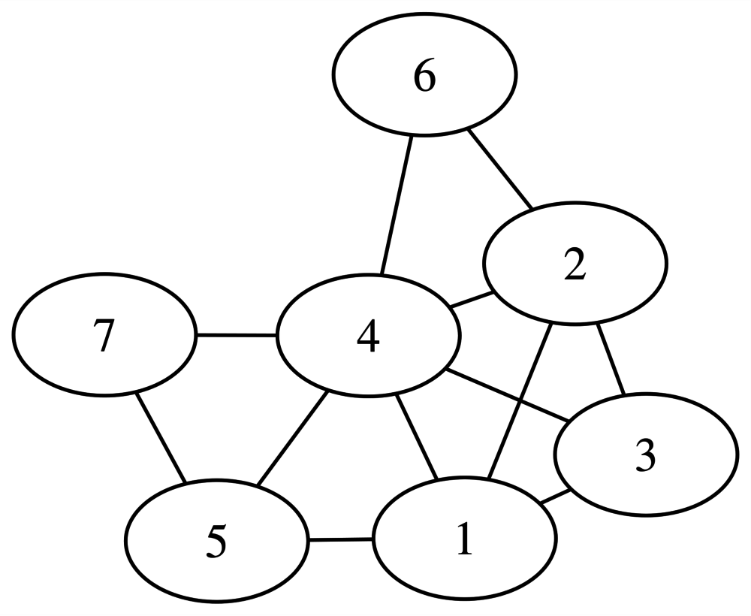


*Figura 1V.* Matriz reducida.

Fuente: Laboratorio IPC2, 2021.

**GraphViz:**

GraphViz es un conjunto de herramientas de software para el diseño de diagramas definido en el lenguaje descriptivo DOT. Con esta herramienta se elaboran diferentes tipos de gráficas como grafos dirigidos, no dirigidos, diagramas de flujo, etc.



*Figura VI.* Grafo.

Fuente: Google Imágenes.

**Conclusiones**

* El concepto de un tipo de dato abstracto es muy interesante, permite manejar información de una manera dinámica y teniendo como límite nuestra imaginación podemos crear muchísimas herramientas muy útiles.
* Hay muchas alternativas para la solución de problemas de NP, cual utilizar y de qué manera gestionarlo depende de nosotros.
* La optimización de peticiones a bases de datos o aplicaciones que manejan bastante información es un tema muy importante para el correcto funcionamiento de todas estas.
* El concepto de matrices se puede aplicar en muchos campos, por lo cual son temas que no debemos dejar pasar desapercibidos, ya que nos pueden servir de muchas formas. Como en el caso de este proyecto.
* Existen muchos lenguajes para el procesamiento de archivos, de los cuales xml es uno de uso muy general y útil en bastantes ámbitos.

**Referencias bibliográficas**

GraphViz (8 de marzo de 2021) En Wikipedia.

<https://es.wikipedia.org/w/index.php?title=Graphviz&oldid=119439121>

Tipo de dato abstracto (8 de marzo de 2021) En Wikipedia.

<https://es.wikipedia.org/w/index.php?title=Tipo_de_dato_abstracto&oldid=131920560>

ElementTree (8 de marzo de 2021) en DataCamp.

<https://www.datacamp.com/community/tutorials/python-xml-elementtree>