VisDB

Visualización de Información

Juan Diego Bejarano

2017079378

Yuberth Elizondo

2016055077

Proyecto 1

13/05/2018

Introducción

En Bases de datos extremadamente grandes, con miles de datos, o hasta millones de datos, normalmente es un problema encontrar los datos, y como están relacionados entre ellos; en sistemas de búsqueda de datos convencionales, como la búsqueda SQL, hasta las personas mas experimentadas en una Base de Datos especifica tienen problemas en encontrar información o más comúnmente encontrar la relación entre 2 o más datos. A través de los años se han hecho una cantidad considerable de acercamientos para mejorar la búsqueda de casos específicos en bases de datos, entre estos tenemos un ejemplo que consiste en hacer una interfaz grafica para ver de manera más fácil los datos (ej. FLEX [Mot 90] o GRADI [KL 92]) o por otro lado tenemos las técnicas que intentan dar un dato aproximado a una búsqueda en específico. El problema de estas técnicas es que utilizan métodos de generalización, por lo que los resultados que nos dan no son cien por ciento confiables. A partir de estos esfuerzos que se hicieron antes, se creo un tipo de búsqueda, en el cual toda la base de datos se la gráfica, no solo se hace una interfaz grafica que ayude a buscar (y no solo en consola como se hacía antes) Haciendo la visualización de datos relacionados son fáciles de ver y se ve de manera agradable los Datos, de aquí nace un nuevo tipo de Técnicas de Visualización como la siguiente.

VisDB es una aplicación que fue creada hace varios años por la "Institute for Computer Science, University of Munich". VisDB utiliza técnicas de visualización de datos, el cual nos deja ver una cantidad grande de datos, normalmente proveniente de Bases de Datos Gigantescas, así se puede ver de manera eficaz, en una pantalla los datos sin perder la profundidad de todos los datos.

La técnica se basa en el uso de pixeles con colores para representar la cantidad de datos sin perder los datos por ser representados en un espacio tan pequeño, es decir una vez utilizada la técnica, el usuario recibe una representación gráfica, fácil de entender de todos los datos.

En nuestra adaptación de VisDB, Basándonos en las ventas de un grupo de vendedores de Walmart realizamos la adaptación de VisDB, utilizamos una técnica el cual se basa en graficar los datos, a partir de un valor de relevancia (en nuestro caso el valor de relevancia puede ser el valor

menor o mayor de una base de datos), se grafica en una manera espiral, siguiendo un patrón claro, luego el color se define por el id del vendedor o valor de las ventas concluidas.

La visualización se realizo en la Herramienta llamada Dioköl, la cual se basa en Lua y Processing.

Visualización

La Técnica de visualización utilizada se basa en el ordenamiento de datos a partir de datos enlazados para así poder encontrar un valor de relevancia que nos sirva para graficar, en el ejemplo que realizamos utilizamos como valor de relevancia el valor menor de la base de datos, la cual consistía en 40096 ventas, entre estos esta el ID del Vendedor, y la Venta realizada en esa instancia.

Para poder utilizar la técnica con otro tipo de bases de datos, la persona debe programar la función linkData y relevanceFactor, las cuales en nuestro caso fueron programadas de la siguiente manera:

```
165    -- Funcion asociar valores, para poder enlazarlos y graficar con iteracciones
166    function linkData(data1,data2)
167        local _table={}
168        for i=1,#data1
169        do
170            local sq={id=data1[i],pair=data2[i]}
171            table.insert(_table,sq)
172        end
173        return _table
174    end
```

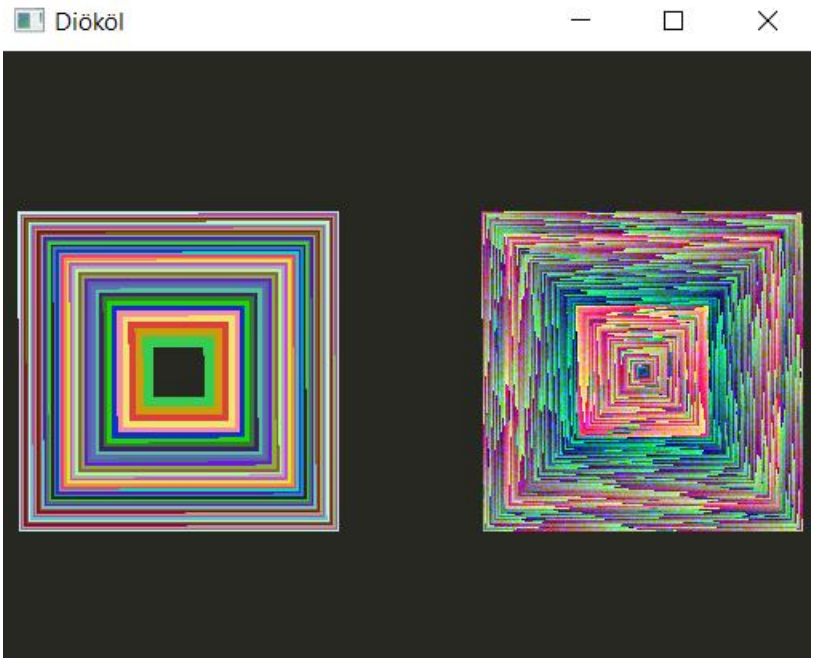
Link data en este caso recibe las dos tablas, el ID del vendedor es data1 y las ventas son data2

```
13    function by_ID(data1, data2)
14        return data1.id < data2.id -- compara por id (1er parametro de la tabla)
15    end
16
17    function relevanceFactor(pTable)
18        table.sort(pTable, by_ID)
19        -- se ordenan los datos de acuerdo a cierto criterio
20        return pTable
21    end
22
```

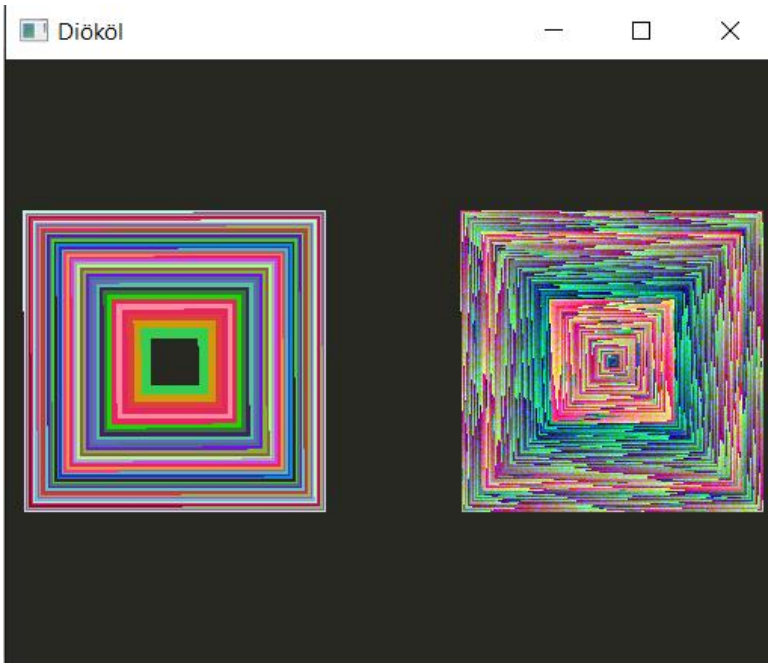
relevanceFactor por otro lado es lo que utilizamos para ordenar los datos antes de dibujarlos, en nuestro caso ordenamos la data enlazada por medio del ID de menor a mayor, así teniendo como valor de relevancia el numero menor de la Base de Datos.

La Base de Datos que utilizamos es un documento extensivo con extensión .csv para un manejo más fácil de los datos dentro de Dioköl.

La visualización realizada se ve de la siguiente manera:



Para el elemento de Interacción, decidimos elegir ver los datos enlazados por medio de la función linkData, realizando un cambio de color, lo cual me lleva al hecho que la interacción es bidireccional, lo que significa esto es que sin importar en cual de los 2 gráficos se haga Click, el cambio de colores se vera reflejados en la otra gráfica. En la siguiente figura hay un ejemplo de datos relacionados por ello cambio de color, la figura anterior es la representación de datos sin cambios en la coloración por la selección de datos.



Conclusiones

A partir de las visualizaciones utilizadas a partir de las técnicas descritas en VisDB llegamos a tener estas graficas donde podemos ver un total de aproximadamente 80000 datos representados en pixeles, haciendo fácil ver la interacción entre los datos.

En la creación de estas técnicas tuvimos varios problemas, entre estos tuvimos como común denominador la Orientación Objetos de Lua, al querer utilizar Objetos para hacer más fácil a los siguientes usuarios de esta aplicación la creación de las funciones linkData y relevanceFactor,

pero tuvimos varios problemas con las tablas de datos en la implementación de objetos, por lo que nos quedamos con la programación imperativa así pudiendo hacer que funcione de manera eficiente. Otro problema es con la herramienta que utilizamos, Dioköl sufría con la representación de tantos datos al mismo tiempo, ya que no podía correr al frameRate predeterminado que se e asigno a la técnica, el cual era 120 FPS, esto no era problema de la computadora que lo corría, ya que lo corrimos en 2 sistema y una con una tarjeta Gráfica NVidia 1050 de 4GB de Memoria GDDDR5, la cual está hecha específicamente para correr datos gráficos en grandes cantidades. En vez de correr a los 120 FPS que le indicamos (es decir hacer 120 ciclos draw de processing en un segundo) hacia 1 ciclo draw cada 2.65 segundos aproximadamente, haciendo la interacción inconsistente y relativamente lenta.

Considerando todos los problemas que tuvimos con la visualización. Fue una representación exitosa de la técnica utilizada por VisDB en su propio sistema de dibujo de Bases de Datos.

Full Code: https://github.com/yelizondo/MultidimensionalVisualization

Referencias Bibliográficas

Keim, D., & Kriegel, H. (1994). VisDB: Database Exploration Using Multidimensional Visualization [Ebook] (1st ed.). München: Institute for Computer Science, University of Munich. Retrieved from https://epub.ub.uni-muenchen.de/4129/1/06.pdf

lua-users wiki: Home Page. (2018). Retrieved from http://lua-users.org/wiki/

Processing Foundation. (2018). Retrieved from https://github.com/processing