Universidad Rafael Landívar
Facultad de Ingeniería
Matemática Discreta II
Ing. Moisés Antonio Alonso Gonzales

PROYECTO NO. 2

ALGORITMO DE DIJKSTRA – Fancy Tours

Francisco Josué Alonzo Chiguichon 1197517

Alejandra Anahí Samayoa Arreaga 1278817

Lester Andrés García Aquino 1003115

Yazmine Isabel Sierra Aragón 1174916

Guatemala 20 de noviembre de 2017

INTRODUCCION

El nacimiento del concepto GRAFOS se puede situar, por el año 1730, cuando Euler (matemático) se convirtió en el padre de la Teoría de Grafos al modelar un famoso problema no resuelto, llamado el "problema de los puentes de Königsberg".

Un grafo G es una dupla G = (X, U), donde X es un conjunto finito y no vacío de elementos llamados vértices y U es el conjunto cuyos elementos se componen de subconjuntos de X de cardinalidad dos (2), llamados aristas.

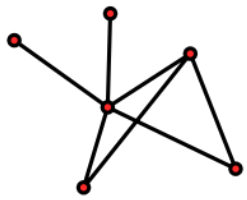
El algoritmo de Dijkstra, también llamado algoritmo de caminos mínimos es un algoritmo para la determinación del camino más corto dado un vértice origen al resto de los vértices en un grafo con pesos en cada arista. Su nombre se refiere a Edsger Dijkstra, quien lo describió por primera vez en 1959.

MARCO TEORICO

¿QUE ES UN GRAFO?

Desde un punto de vista práctico, los grafos permiten estudiar las interrelaciones entre unidades que interactúan unas con otras. Por ejemplo, una red de computadoras puede representarse y estudiarse mediante un grafo, en el cual los vértices representan terminales y las aristas representan conexiones (las cuales, a su vez, pueden ser cables o conexiones inalámbricas).

Grafo simple



TIPOS DE GRAFOS

1. Grafo simple: O simplemente grafo es aquel que acepta una sola arista uniendo dos vértices cualesquiera. Esto es equivalente a decir que una arista cualquiera es la única que une dos vértices específicos. Es la definición estándar de un grafo.

2. Multígrafo: Es el que acepta más de una arista entre dos vértices. Estas aristas se llaman múltiples o lazos (*loops* en inglés). Los grafos simples son una subclase de esta categoría de grafos. También se les llama grafos generales.

3. Pseudografo: Si incluye algún lazo.

4. Grafo orientado: grafo dirigido o dígrafo. Son grafos en los cuales se ha añadido una orientación a las aristas, representada gráficamente por una flecha.

5. Grafo etiquetado: Grafos en los cuales se ha añadido un peso a las aristas (número entero generalmente) o un etiquetado a los vértices.

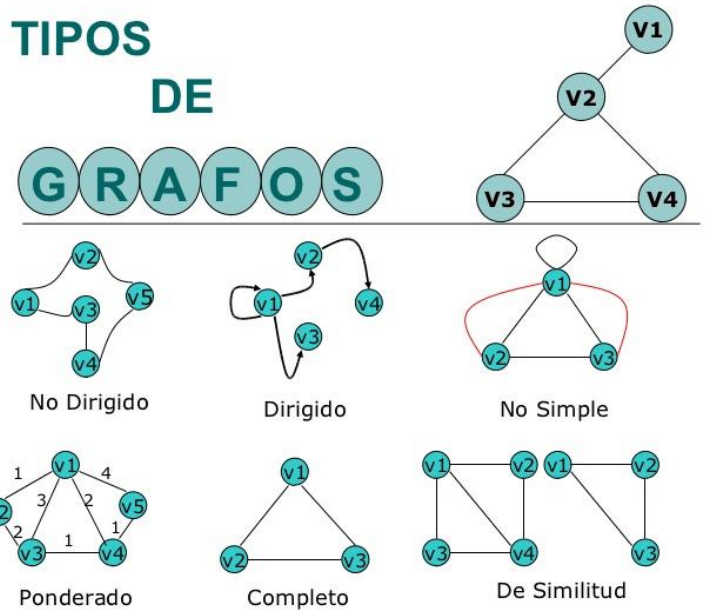
6. Grafo aleatorio: Grafo cuyas aristas están asociadas a una probabilidad.

7. Hipogrifo: Grafos en los cuales las aristas tienen más de dos extremos, es decir, las aristas son incidentes a 3 o más vértices.

8. Grafo infinito: Grafos con conjunto de vértices y aristas de cardinal infinito.

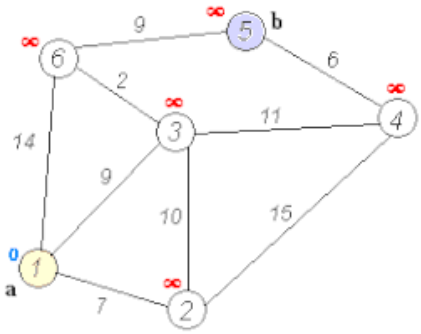
9. Grafo plano: Los grafos planos son aquellos cuyos vértices y aristas pueden ser representados sin ninguna intersección entre ellos. Podemos establecer que un grafo es plano gracias al Teorema de Kuratowski.

10. Grafo regular: Un grafo es regular cuando todos sus vértices tienen el mismo grado de valencia.



ALGORITMO DE DIJKSTRA

La idea subyacente en este algoritmo consiste en ir explorando todos los caminos más cortos que parten del vértice origen y que llevan a todos los demás vértices; cuando se obtiene el camino más corto desde el vértice origen, al resto de vértices que componen el grafo, el algoritmo se detiene. El algoritmo es una especialización de la búsqueda de costo uniforme, y como tal, no funciona en grafos con aristas de coste negativo (al elegir siempre el nodo con distancia menor, pueden quedar excluidos de la búsqueda nodos que en próximas iteraciones bajarían el costo general del camino al pasar por una arista con costo negativo**).**



PASOS PARA EL ALGORIMO DE DIJKSTRA

1. Inicializar todas las distancias en D con un valor infinito relativo ya que son desconocidas al principio, exceptuando la de x que se debe colocar en 0 debido a que la distancia de x a x sería 0.

2. Sea a = x (tomamos a como nodo actual).

3. Recorremos todos los nodos adyacentes de a, excepto los nodos marcados, llamaremos a estos nodos no marcados vi.

4. Para el nodo actual, calculamos la distancia tentativa desde dicho nodo a sus vecinos con la siguiente fórmula: dt(vi) = Da + d(a,vi). Es decir, la distancia tentativa del nodo 'vi' es la distancia que actualmente tiene el nodo en el vector D más la distancia desde dicho nodo 'a' (el actual) al nodo vi. Si la distancia tentativa es menor que la distancia almacenada en el vector, actualizamos el vector con esta distancia tentativa. Es decir: Si dt(vi) < Dvi → Dvi = dt(vi).

5. Marcamos como completo el nodo a.

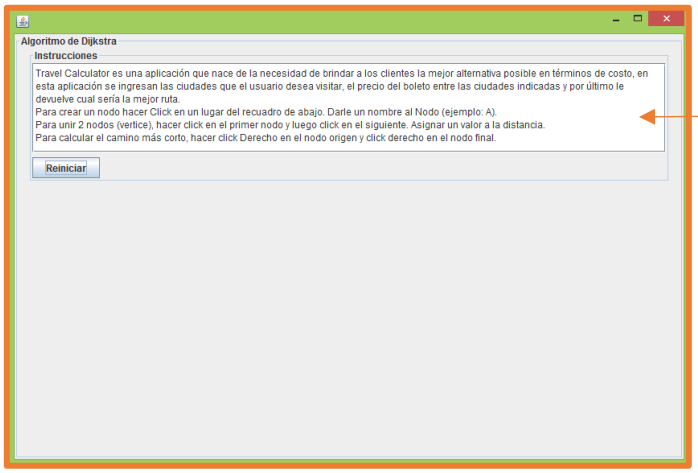
6. Tomamos como próximo nodo actual el de menor valor en D (puede hacerse almacenando los valores en una cola de prioridad) y volvemos al paso 3 mientras existan nodos no marcados.

7. Una vez terminado al algoritmo, D estará completamente lleno

PSEUDOCODIGO DEL ALGORITMO DE DIJKSTRA

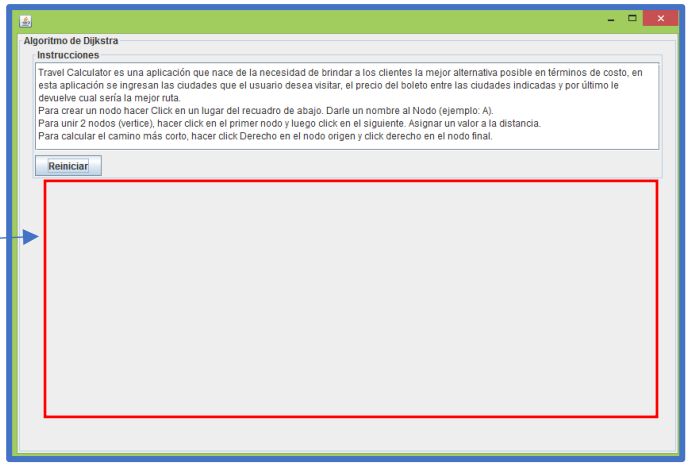
```
DIJKSTRA (Grafo G, nodo_fuente s)
    para u ∈ V[G] hacer
        distancia[u] = INFINITO
        padre[u] = NULL
        visto[u] = false
    distancia[s] = 0
    adicionar (cola, (s, distancia[s]))
    mientras que cola no es vacía hacer
        u = extraer.mínimo(cola)
        visto[u] = true
        para todos v ∈ adyacencia[u] hacer
            si no visto[v] y distancia[v] > distancia[u] + peso (u, v) hacer
                distancia[v] = distancia[u] + peso (u, v)
                padre[v] = u
                adicionar(cola,(v, distancia[v]))
```

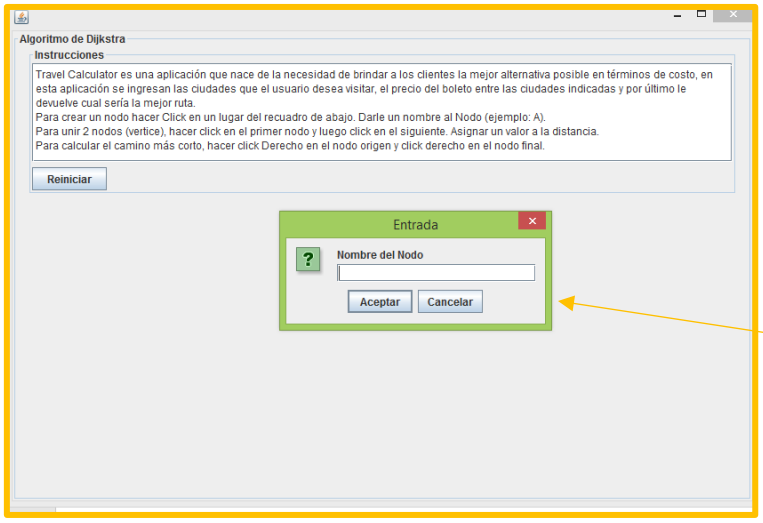
MANUAL DE USUARIO



Ventana de Principal, en la cual muestra las instrucciones para utilizar el programa

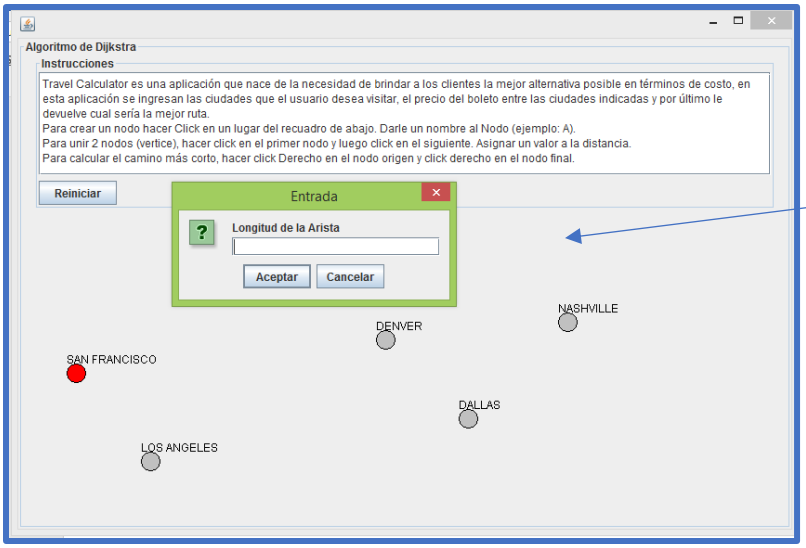
Para la inserción de nodos, dar clic en cualquier parte del panel (lo que está marcado de color rojo).



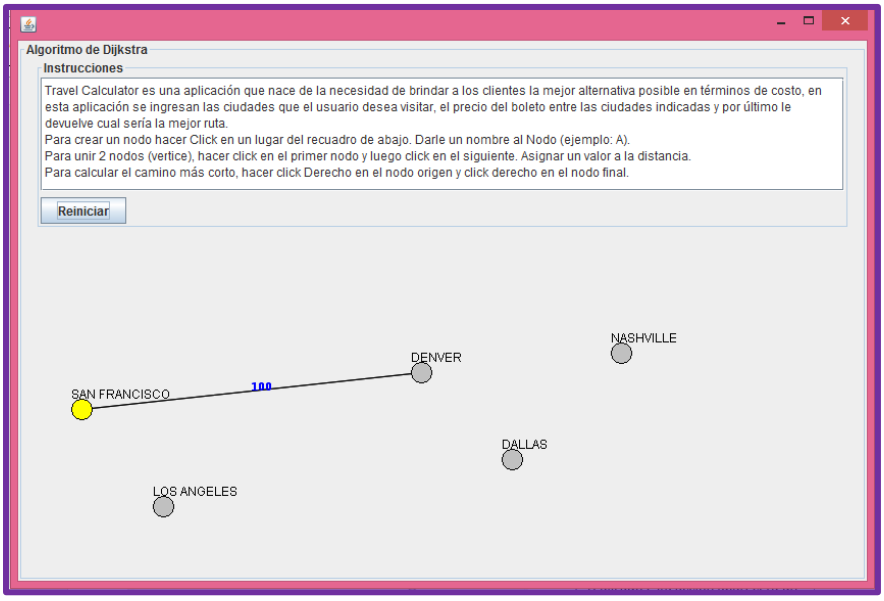


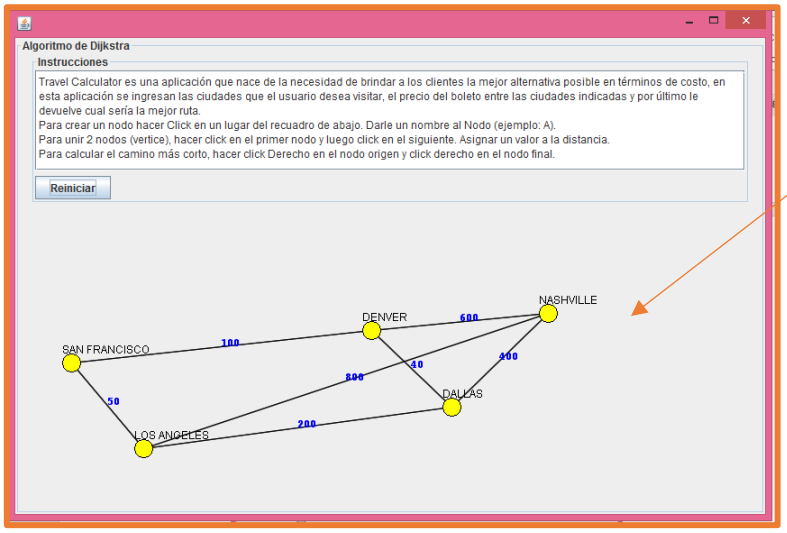
Al haber dado clic en cualquier parte del panel, aparecerá una ventana llamada “entrada”, la cual pide el ingreso del nombre del Nodo.

Y así se repite el proceso con cualquier nodo que se desee insertar

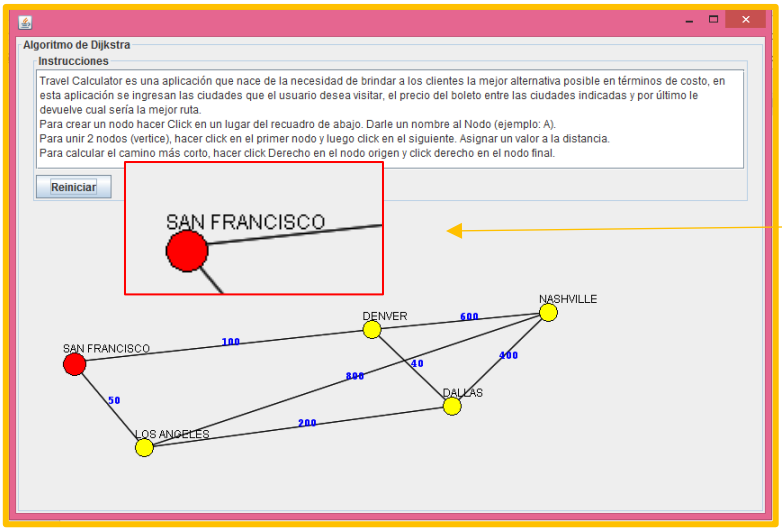


Para la unión de nodos por medio de aristas, se debe dar clic al Primer Nodo, en este caso sería “San Francisco”, luego hacer clic al segundo Nodo, llamado “Denver”. Después de haber realizado esta acción aparecerá una ventana para el ingreso de la longitud de arista entre San Francisco y Denver



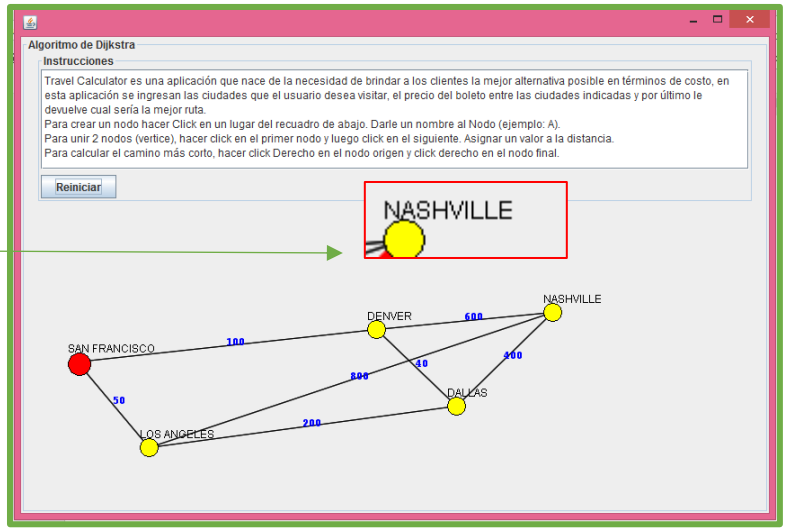


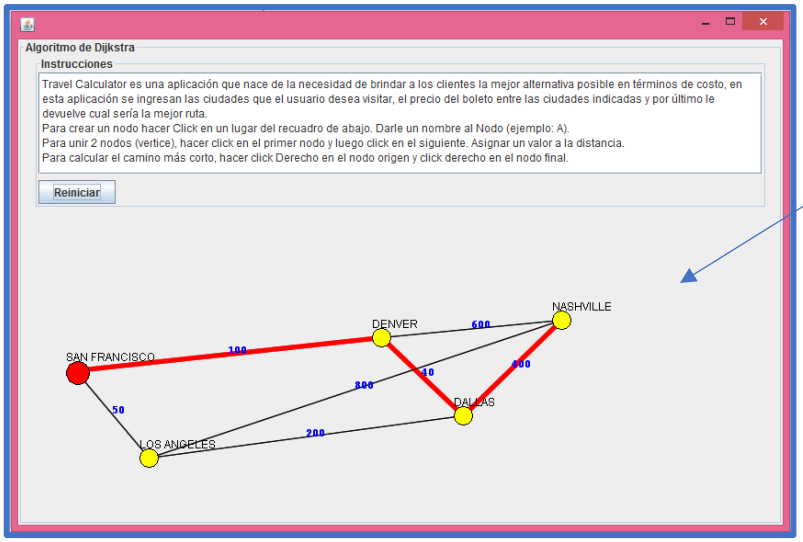
Después de haber unido cada Nodo con su respectiva arista, deberá crearse un grafo.



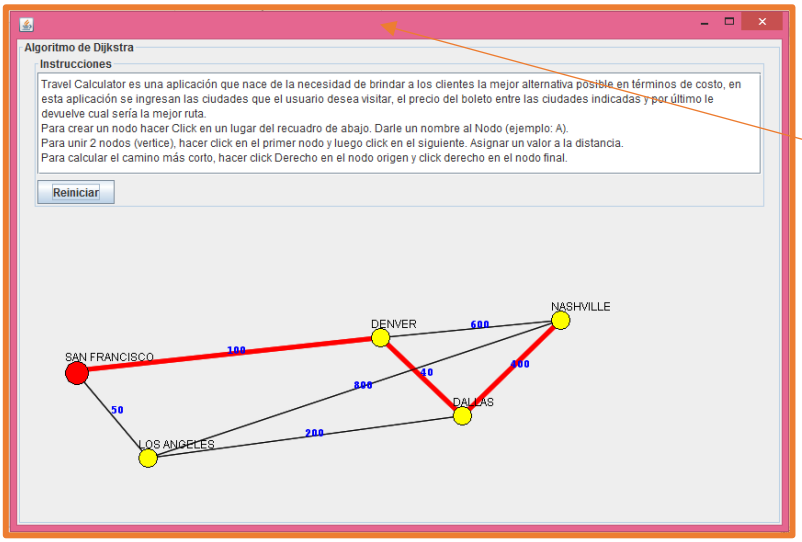
Para conocer el camino recorrido mas corto entre el nodo origen y el nodo destino se hará clic derecho en el Nodo Origen “San Francisco”.

Luego clic derecho en el nodo destino “Nashville”.





Automáticamente se marcarán las aristas de color rojo, las cuales hacen el recorrido más corto.



Se encuentra la opción “Reiniciar”, para la creación de un grafo nuevo.

CONCLUSIONES

1. Existen problemas que a simple vista no parecieran poder modelarse como una red, pero una vez logrado tal modelamiento, es posible resolverlos con los algoritmos mencionados a lo largo del presente trabajo investigativo.

2. Para determinar la distancia más corta entre un nodo y otro se empleará el algoritmo de Dijkstra.

3. A nuestro punto de vista estos temas son importantes para el área de sistemas computacionales ya que en relaciones son orden y divisibilidad entre números, las relaciones de equivalencia entre los datos de entrada de un programa en cuanto a la detección de posibles errores de programación, la relación de dependencia entre las distintas fases producción en una industria o la agrupación de datos aislados en complejas bases de datos con relaciones de dependencia entre sus campos

4. Gracias a esta aplicación, se podrá encontrar la ruta más óptima para los viajeros.

5. Para la creación de este programa se implementó el algoritmo Dijkstra.

6. Se pudo observar que este algoritmo siempre devuelve la ruta más corta.

7. El tipo de grafo que se utiliza es el ponderado, ya que cada camino posee un peso.