Algoritmo de Kruskal

Luis Enrique López Nerio Universidad Autónoma de Nuevo León
luiselopeznerio@gmail.com

25 de Octubre del 2017

Abstract

Para este reporte se cubrirá otro tema del área de teoría de grafos, hasta el momento hemos visto diversos algoritmos que nos ayudan a encontrar información importante dentro de un grafo, nuestro siguiente objetivo será dado un grafo, encontrar todas las artistas del grafo que conecten a todos los vértices y que la suma de estos pesos sea mínima.
Para realizar esta tarea se introducirá el algoritmo de Kruskal, además se mencionaran algunas aplicaciones para este algoritmo.

I. El arbol de expansión minima

UUn grafo es un objeto matemático que contiene vértices y aristas, en si se puede definir matemáticamente como un dos conjuntos, el de vértices y el de aristas. El grafo puede ser ponderado o no ponderado, si es ponderado entonces a cada arista le corresponde un peso. Todas estas definiciones ya se introdujeron y se explicaron más a detalle en reportes anteriores.

Ahora se nos presentara un problema, que pasa si yo tengo un grafo, de este grafo que llamaremos original yo quiero un subgrafo, este subgrafo debe tener los mismo vértices que mi grafo original y debe tener aristas de tal manera que todos los vértices queden conectados pero sin que se presenten ciclos, esto es, del grafo original encuentra un grafo que contenga todas las aristas que conectan a todos los vértices y que la suma de los pesos sea mínima.

Este grafo que se genera del grafo original se llama árbol de expansión mínima, en la figura 1 se puede ver un ejemplo, las aristas resaltadas conectan todos los vértices del grafo original y la suma de las aristas es la mínima comparada con otros árboles de expansión para este mismo grafo. Ahora que definimos el problema, surge la pregunta, Â£Cómo encuentro el árbol de expansión minima de un grafo?, para responder esta pregunta introduciremos el algoritmo de Kruskal

II. Algoritmo de Kruskal

El algoritmo de Kruskal es un algoritmo que encuentra el arbol de expansión minima de un grafo, fue publicado por primera vez por Joseph Kruskal en 1956. La idea principal de este algoritmo es:

1 Se crea un grafo F, donde cada vertice del grafo es un árbol diferente
2 Se crea un conjunto S que contenga todos las aristas del grafo original
3 Mientras S no este vacio y F no conecte a todos los vertices
3 Removemos la arista con peso minimo de S
4 Si la arista removida conecta dos difentes arboles de F encontes agregamos la arista a F, de esta manera dos arboles separados forman uno.

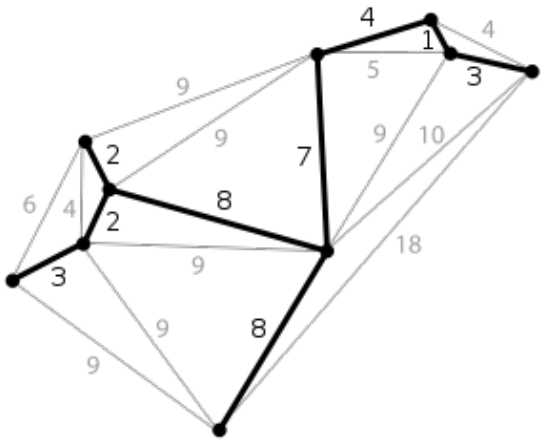

Figure 1: *ejemplo de un arbol de expansión minima*

A continuación se presenta el codigo en python para el algoritmo de Kruskal

```
def kruskal(self):
    e = deepcopy(self.E)
    arbol = grafo()
    peso = 0
    comp = dict()
    t = sorted(e.keys(), key = lambda k: e[k], reverse=True)
    nuevo = set()
    while len(t) > 0 and len(nuevo) < len(self.V):
        #print(len(t))
        arista = t.pop()
        w = e[arista]
        del e[arista]
        (u,v) = arista
        c = comp.get(v, {v})
        if u not in c:
            #print('u ',u, 'v ',v ,'c ', c)
            arbol.conecta(u,v,w)
            peso += w
            nuevo = c.union(comp.get(u,{u}))
            for i in nuevo:
                comp[i]= nuevo
    print('MST con peso', peso, ':', nuevo, '\n', arbol.E)
    return arbol
```

Codigo 1. Algoritmo de Kruskal

Como podemos ver este algoritmo regresa como resultado el arbol de expansión minima.

III. Experimento

Para probar nuestro algoritmo de Kruskal se realizara un experimento, de la ciudad de Monterrey tomaremos 10 lugares turisticos, estos lugares turisticos seran considerados los vertices de un grafo, de cada lugar turistico calcularemos el tiempo que nos toma en promedio llegar a cada unos de los otros, el tiempo promedio entre dos lugares sera el peso de la arista que los une.

Los 10 lugares que eligi son los siguientes:

- Cola de Caballo
- Faro de Comercio
- Chipinque
- Grutas de Garcia
- Fundidora
- Cadereyta
- Basilica
- Cerro de la Silla
- Cerro de las Mitras
- Estadio BBVA

Ahora, la idea es encontrar un camino que comienze en alguno de estos vertices y que se vaya moviendo a cada uno de los otros sin pasar por uno que ya haya visitado, para finalmente llegar al vertice inicial, este camino también recibe el nombre de Ciclo Hamiltoniano.

Naturalmente se encontraran diferentes ciclos en el grafo, lo importante es tratar de encontrar el que nos haga pasar por aristas de tal manera que la suma de los pesos de las aristas visitadas sea la menor posible. Para realizar esta tarea utilizaremos nuestro algoritmos de Kruskal ademas del siguiente codigo:

```
  m = grafo()
m.conecta('Cola de Caballo', 'Faro de Comercio', 46)
m.conecta('Cola de Caballo', 'Chipinque', 69)
m.conecta('Cola de Caballo', 'Grutas de Garcia', 91)
m.conecta('Cola de Caballo', 'Fundidora', 47)
m.conecta('Cola de Caballo', 'Cadereyta', 63)
m.conecta('Cola de Caballo', 'Basilica', 48)
m.conecta('Cola de Caballo', 'Cerro de la Silla', 61)
m.conecta('Cola de Caballo', 'Cerro de las Mitras', 66)
m.conecta('Cola de Caballo', 'Estadio BBVA', 50)

m.conecta('Faro de Comercio','Chipinque', 30)
m.conecta('Faro de Comercio','Grutas de Garcia', 49)
m.conecta('Faro de Comercio','Fundidora', 7)
m.conecta('Faro de Comercio','Cadereyta', 38)
m.conecta('Faro de Comercio','Basilica', 5)
m.conecta('Faro de Comercio','Cerro de la Silla', 24)
m.conecta('Faro de Comercio','Cerro de las Mitras',23)
m.conecta('Faro de Comercio','Estadio BBVA', 13)

m.conecta('Chipinque','Grutas de Garcia',66 )
m.conecta('Chipinque','Fundidora', 30)
m.conecta('Chipinque','Cadereyta', 62)
m.conecta('Chipinque','Basilica', 28)
m.conecta('Chipinque','Cerro de la Silla', 48)
m.conecta('Chipinque','Cerro de las Mitras', 41)
m.conecta('Chipinque','Estadio BBVA', 37)

m.conecta('Grutas de Garcia','Fundidora', 51)
m.conecta('Grutas de Garcia','Cadereyta', 81)
m.conecta('Grutas de Garcia','Basilica',49 )
m.conecta('Grutas de Garcia','Cerro de la Silla', 68)
m.conecta('Grutas de Garcia','Cerro de las Mitras', 48)
m.conecta('Grutas de Garcia','Estadio BBVA', 56)

m.conecta('Fundidora','Cadereyta', 34)
m.conecta('Fundidora','Basilica',8 )
m.conecta('Fundidora','Cerro de la Silla', 22)
```

```
m.conecta('Fundidora','Cerro de las Mitras', 23)
m.conecta('Fundidora','Estadio BBVA', 8)

m.conecta('Cadereyta','Basilica', 38)
m.conecta('Cadereyta','Cerro de la Silla', 43)
m.conecta('Cadereyta','Cerro de las Mitras', 56)
m.conecta('Cadereyta','Estadio BBVA', 34)

m.conecta('Basilica','Cerro de la Silla', 26)
m.conecta('Basilica','Cerro de las Mitras', 25)
m.conecta('Basilica','Estadio BBVA', 15 )

m.conecta('Cerro de la Silla','Cerro de las Mitras', 45)
m.conecta('Cerro de la Silla','Estadio BBVA', 19)

m.conecta('Cerro de las Mitras','Estadio BBVA', 32)




k = m.kruskal()
for r in range(50):
    ni = random.choice(list(k.V))
    dfs =  k.DFS(ni)
    c = 0
    #print(dfs)
    #print(len(dfs))
    for f in range(len(dfs) -1):
            c += m.E[(dfs[f],dfs[f+1])]
            print(dfs[f], dfs[f+1], m.E[(dfs[f],dfs[f+1])] )

    c += m.E[(dfs[-1],dfs[0])]
    print(dfs[-1], dfs[0], m.E[(dfs[-1],dfs[0])])
    print('costo',c,'\n')
```

Codigo 2. Codigo para encontrar el ciclo menor

Al correr el codigo nos encontramos que el camino con el costo menor es el siguiente: Cadereyta a Estadio BBVA 34 Estadio BBVA a Fundidora 8 Fundidora a Cerro de las Mitras 23 Cerro de las Mitras a Grutas de Garcia 48 Grutas de Garcia a Faro de Comercio 49 Faro de Comercio a Basilica 5 Basilica a Chipinque 28 Chipinque a Cola de Caballo 69 Cola de Caballo a Cerro de la Silla 61 Cerro de la Silla a Cadereyta 43

costo 368 minutos

Entonces el camino que recorre todos los vertices y que tiene un tiempo menor comienza en Cadereyta y le toma 6 horas y 8 minutos visitar cada uno de los vertices.

IV. Conclusión

Mi conclusión es que el algoritmo de Kruskal nos da un abanico de posibilidades en las cuales podemos aplicar nuestra clase de Grafos, a su vez el experimento se puede mejorar ya que utliza el tiempo promedio que toma viajar de nodo a nodo sin tomar en cuenta otras caracteristicas como la hora del dia o si los caminos utilizados pasan por carreteras con peaje.