Jorge Adrián Padilla Velasco
A00570894
17 de septiembre de 2018

Algoritmos de Grafos

A* (A asterisco)

Es un algoritmo de búsqueda que puede ser empleado para el cálculo de caminos mínimos en una red. Se va a tratar de un algoritmo heurístico, ya que una de sus principales características es que hará uso de una función de evaluación heurística, mediante la cual etiquetará los diferentes nodos de la red y que servirá para determinar la probabilidad de dichos nodos de pertenecer al camino óptimo.

Esta función de evaluación que etiquetará los nodos de la red estará compuesta a su vez por otras dos funciones. Una de ellas indicará la distancia actual desde el nodo origen hasta el nodo a etiquetar, y la otra expresará la distancia estimada desde este nodo a etiquetar hasta el nodo destino hasta el que se pretende encontrar un camino mínimo. Es decir, si se pretende encontrar el camino más corto desde el nodo origen s, hasta el nodo destino t, un nodo intermedio de la red n tendría la siguiente función de evaluación f(n) como etiqueta:

f(n) = g(n) + h(n)

Donde:

g(n) indica la distancia del camino desde el nodo origen s al n.

h(n) expresa la distancia estimada desde el nodo n hasta el nodo destino t.

Pseudocódigo:

- 1.- Establecer el nodo s como origen. Hacer f(s)=0, y f(i)=∞ para todos los nodos i diferentes del nodo s. Iniciar el conjunto Q vacío.
- 2.- Calcular el valor de f(s) y mover el nodo s al conjunto Q .
- 3.- Seleccionar el nodo i del conjunto Q que presente menor valor de la función f(i) y eliminarlo del conjunto Q.
- 4.- Analizar los nodos vecinos j de i. Para cada enlace (i, j) con coste cij hacer:
    - 4.1.-Calcular: f(j)'=g(i)+cij +h(j)
    - 4.2.-Si f(j)'<f(j),
        - 4.2.1.-Actualizar la etiqueta de j y su nuevo valor será: f(j)= g(i)+cij +h(i)
        - 4.2.2.-Insertar el nodo j en Q
    - 4.3.-Si f(j)'≥f(j)
        - 4.3.1.-Dejar la etiqueta de j como está, con su valor f(j)
- 5.- Si Q está vacío el algoritmo se termina. Si no está vacío, volver al paso 3.

Complejidad:

En el peor de los casos, con una heurística de pésima calidad, la complejidad será exponencial, mientras que en el mejor de los casos, con una buena heurística, el algoritmo se ejecutará en tiempo lineal.

BFS (Breadth First Search)

Es una forma de encontrar todos los vértices alcanzables de un grafo partiendo de un vértice origen dado. Como en el algoritmo de búsqueda en profundidad, BFS recorre una componente conexa de un grafo y define un árbol de expansión.

Intuitivamente, la idea básica de la búsqueda en anchura es la siguiente: lanzar una ola desde el origen s. La ola golpea a todos los vértices situados a una distancia de una arista de s. Desde allí, la ola golpea a todos los vértices situados a una distancia de dos aristas de s, y así sucesivamente.

Este algoritmo de grafos es muy útil en diversos problemas de programación. Por ejemplo halla la ruta más corta entre dos vértices cuando el peso entre todos los nodos es 1, cuando se requiere llegar con un movimiento de caballo de un punto a otro con el menor número de pasos, o para salir de un laberinto con el menor número de pasos, etc.

Pseudocódigo:

```
BFS (G, s):
 Visitados[s] ← true,    ∀ u ≠ s Visitados[s] ← false
 L[0] ← L[0] ∪ {s},    i ← 0,    T ← Ø
 while L[i] ≠ Ø do
    L[i+1] ← Ø
    foreach u in L[i] do
       foreach v in G do
          if G.Matriz[u,v] ≠0 & Visitados[v] = false then
                Visitados[v] ← true
             T ← T ∪ {(u, v)}
             L[i+1] ← L[i+1] ∪ {v}
          endif
       endfor
    endfor
    i ← i + 1
 endwhile
```

Complejidad:

$O(|V|^2)$ sinn listas de adyacencia

O(|V| + |E|) con listas de adyacencia.

DFS (Depth first Search)

Es un algoritmo que permite recorrer todos los nodos de un grafo o árbol (teoría de grafos) de manera ordenada, pero no uniforme. Su funcionamiento consiste en ir expandiendo todos y cada uno de los nodos que va localizando, de forma recurrente, en un camino concreto. Cuando ya no quedan más nodos que visitar en dicho camino, regresa (Backtracking), de modo que repite el mismo proceso con cada uno de los hermanos del nodo ya procesado.

Pseudocódigo:

```
DFS (G, s):
 ∀ i ∈ V, Visitados[i] ← false
 push (S, s)
 while S ≠ Ø do
   u ← pop (S)
   if Visitados[u] = false then
     Visitados[u] ← true
     foreach v in G.Adyacencia[u] do
       push (S, v)
     endfor
   endif
 endwhile
```

Complejidad:

- Añadir y borrar un no de S, O(1)
- El número de operaciones en S acotado por el grado de G, O(|E|)
- Operaciones en la lista acotadas por |V|
- O(|E| + |V|) utilizando listas de Adyacencia

Problema del agente viajero (TSP)

Es considerado como un conjunto de grafos cuyas aristas son los posibles caminos que puede seguir la entidad para visitar todos los nodos. Es un problema donde intervienen cierto número de variables donde cada variable puede tener N diferentes valores y cuyo número de combinaciones es de carácter exponencial, lo que da lugar a múltiples soluciones óptimas.

Pseudocódigo:

Definir el número de nodos, su posición y el costo por cada arista (i, j) donde i = ciudad 1 y j = ciudad 2

Elegir el nodo inicial i

Hacer

Si el nodo más cercano no se ha visitado

Visitar nodo j

Actualizar lista de nodos visitados

Costo_total = costo_total + costoij

Nodo i = nodo j

Hasta haber visitado todos los nodos

Complejidad:

La solución mas directa puede ser intentar todas las permutaciones (combinaciones ordenadas) y ver cuál de estas es la menor (usando una búsqueda de fuerza bruta). El tiempo de ejecución es un factor polinómico de orden O(n!), el factorial del número de ciudades, esta solución es impracticable para dado solamente 20 ciudades.

Referencias

http://idelab.uva.es/algoritmo

https://es.wikipedia.org/wiki/Algoritmo_de_b%C3%BAsqueda_A%2A#Complejidad_computacional

http://www.dma.fi.upm.es/personal/gregorio/grafos/web/iagraph/busqueda.html

http://alejandromoran.com/wp-content/uploads/2012/11/3-Grafos.pdf

http://inteligencia7b.blogspot.com/2010/11/depth-first-search-primera-profundidad.html

https://www.uaeh.edu.mx/scige/boletin/tlahuelilpan/n3/e5.html

https://es.wikipedia.org/wiki/Problema_del_viajante