Universidad de La Laguna

Analísis de Floyd-Warshall

Diseño y análisis de algoritmos

Carlos Troyano Carmona
Miguel Bravo Arvelo

Universidad de La Laguna

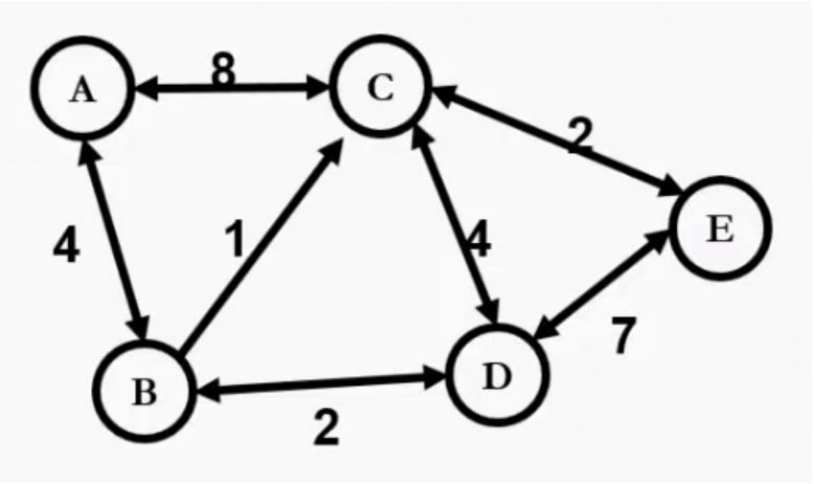
Teoria de grafos :

Caminos mínimos

El problema del camino mínimo

En teoría de grafos el problema del camino más corto consiste en encontrar un camino entre dos vértices (o nodos). Dirigidos, no dirigidos o mixtos pueden distinguirse en las siguientes variantes:

- Camino mínimo de un único origen
- Camino mínimo a una única destinación
- Camino mínimo de todos lo pares de nodos



Universidad de La Laguna

Algoritmo de :

Floyd-Warshall

Floyd-Warshall

El algoritmo de Floyd - Warshall compara todos los posibles caminos entre cada par de vértices.

Teniendo este algoritmo el objetivo es encontrar el camino mínimo de cada nodo i hasta j .

Matriz de distancias

	A	B	C	E	F
A	0	4	8	∞	∞
B	4	0	1	2	∞
C	8	∞	0	4	2
D	∞	2	4	0	7
E	∞	∞	2	7	0

Matriz de recorridos

	A	B	C	E	F
A	-	B	C	E	F
B	A	-	C	E	F
C	A	B	-	E	F
D	A	B	C	-	F
E	A	B	C	E	-

Floyd-Warshall

El algoritmo funciona de tal manera que computa**(i,j,k)** para todos los pares **(i,j)** para k=1, k=2, etc. Hasta k=N, y habiendo hallado el camino mínimo para todos los pares **(i,j)**.

Matriz de distancias

	A	B	C	E	F
A	0	4	8	∞	∞
B	4	0	1	2	∞
C	8	∞	0	4	2
D	∞	2	4	0	7
E	∞	∞	2	7	0

Matriz de recorridos

	A	B	C	E	F
A	-	B	C	E	F
B	A	-	C	E	F
C	A	B	-	E	F
D	A	B	C	-	F
E	A	B	C	E	-

Floyd-Warshall

Universidad de La Laguna

Pseudocodigo

```
1 let dist be a |V| × |V| array of minimum distances initialized to ∞ (infinity)
2 for each edge (u,v)
3    dist[u][v] ← w(u,v)  // the weight of the edge (u,v)
4 for each vertex v
5    dist[v][v] ← 0
6 for k from 1 to |V|
7    for i from 1 to |V|
8       for j from 1 to |V|
9          if dist[i][j] > dist[i][k] + dist[k][j]
10             dist[i][j] ← dist[i][k] + dist[k][j]
11         end if
```

Floyd-Warshall

Una vez el algoritmo ha dado la solución se puede reconstruir el camino usando la matriz de recorridos para cualquier par**(i,j)**.

Matriz de distancias

	A	B	C	E	F
A	0	4	5	6	7
B	4	0	1	2	3
C	8	6	0	4	2
D	6	2	3	0	5
E	10	8	2	6	0

Matriz de recorridos

	A	B	C	E	F
A	-	B	B	B	C
B	A	-	C	D	C
C	A	D	-	D	E
D	B	B	B	-	C
E	C	B	C	C	-

Ej : D -> B -> A

Floyd-Warshall

Universidad de La Laguna

Pseudocodigo

```
1 procedure Path(u, v)
2   if next[u][v] = null then
3       return []
4   path = [u]
5   while u ≠ v
6       u ← next[u][v]
7       path.append(u)
8   return path
```

Universidad de La Laguna

Algoritmo de :

Dijkstra's

Dijkstra's

Planteamiento

- Un grafo de **N** Nodos y **V** vértices
- Problema de máximos / mínimos
- Camino más corto entre el nodo **v** y todos los nodos
- EL planteamiento básico
    - Si tenemos un tres nodos **A**, **B** y **C** el camino más corto entre **A** y **B** ¿Pasa el camino pasa por **C?**
    - Para saber el camino más corto de un nodo cualquiera **v** a otro **w** tenemos que saber si el camino más corto pasa por cualquier nodo N(sub k)

Dijkstra's

Complejidad

- Contar Operaciones
- Buscar nodos adyacentes al conjunto definitivo se ejecuta **n -1** veces
- Este proceso se ejecuta como máximo **n-1** veces
- Las demás operaciones la suma y al comparación no aumentan la complejidad
- **$O(N^2)$**

Dijkstra's

Problema con los ciclos negativos y mejoras

- Nunca terminará
- Una pila de fibonacci o pila binaria mejora su eficacia
- **O(|A| log |V|)**

Dijkstra's

Pseudocodigo

```
1  function Dijkstra(Graph, source):
2      create vertex set Q
3
4      for each vertex v in Graph:
6          dist[v] ← INFINITY
7          prev[v] ← UNDEFINED
8          add v to Q
9      dist[source] ← 0
10
11     while Q is not empty:
13         u ← vertex in Q with min dist[u]
14
15         remove u from Q
16
17         for each neighbor v of u:
18             alt ← dist[u] + length(u, v)
19             if alt < dist[v]:
20                 dist[v] ← alt
21                 prev[v] ← u
23     return dist[], prev[]
```

Universidad de La Laguna

Universidad de La Laguna

Analísis

Experimentales

Pruebas realizadas

Parámetros introducidos en las pruebas:

- Cada nodo tiene un número aleatorio de arcos.
- Todos los arcos tienen un valor positivo de valor máximo 20.
- El número de arcos corresponde al doble del número de nodos.
- En cada prueba se incrementa el número de nodos por el doble.

Analisis experimentales

Universidad de La Laguna

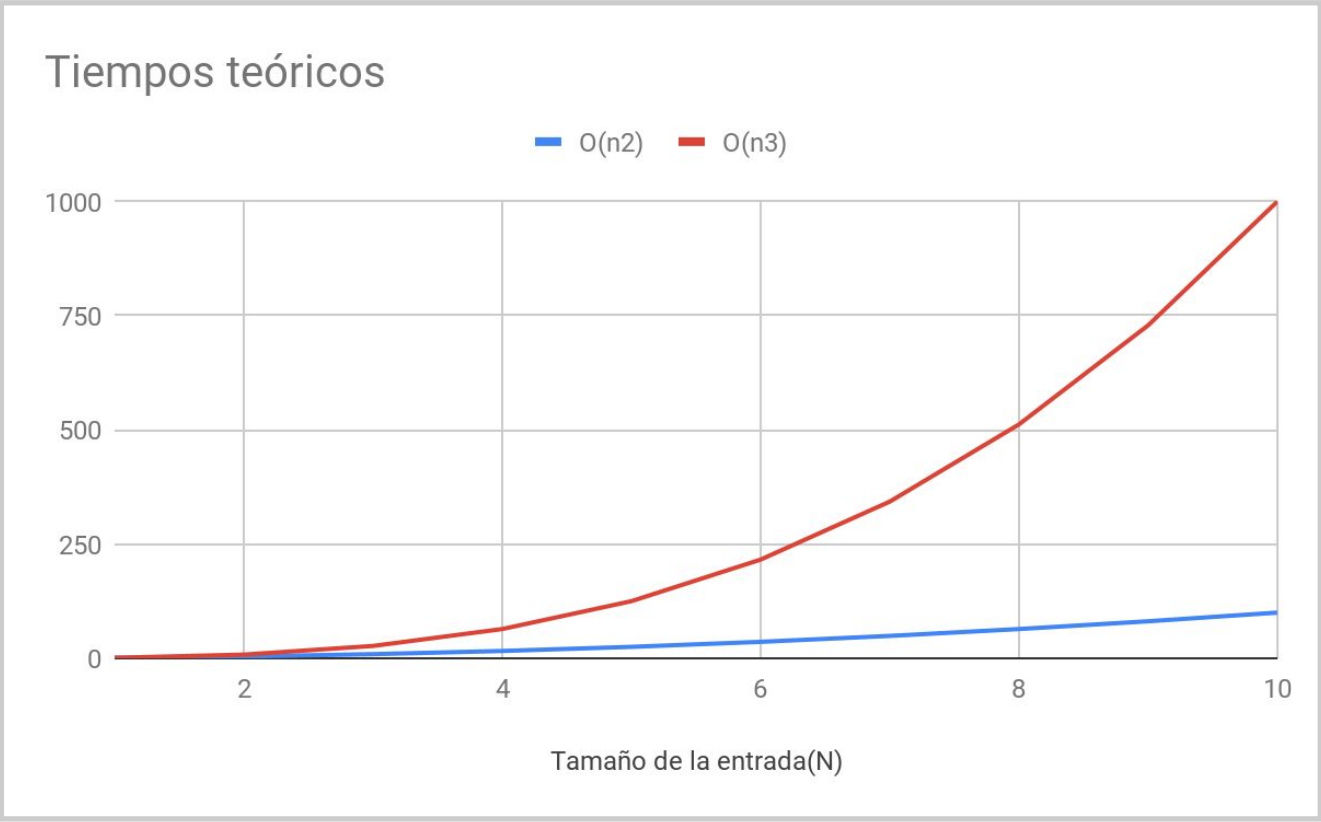
Pruebas realizadas

Arcos	Nodos	Tiempo(ms) Floyd-Warshall	Tiempo(ms) Dijkstra
10	5	0	0
20	10	0	0
40	20	0	1
80	40	2	6
160	80	2	9
320	160	11	33
640	320	23	143
1280	640	170	1062
2560	1280	1518	8214
5120	2560	12195	75948

Analisis experimentales

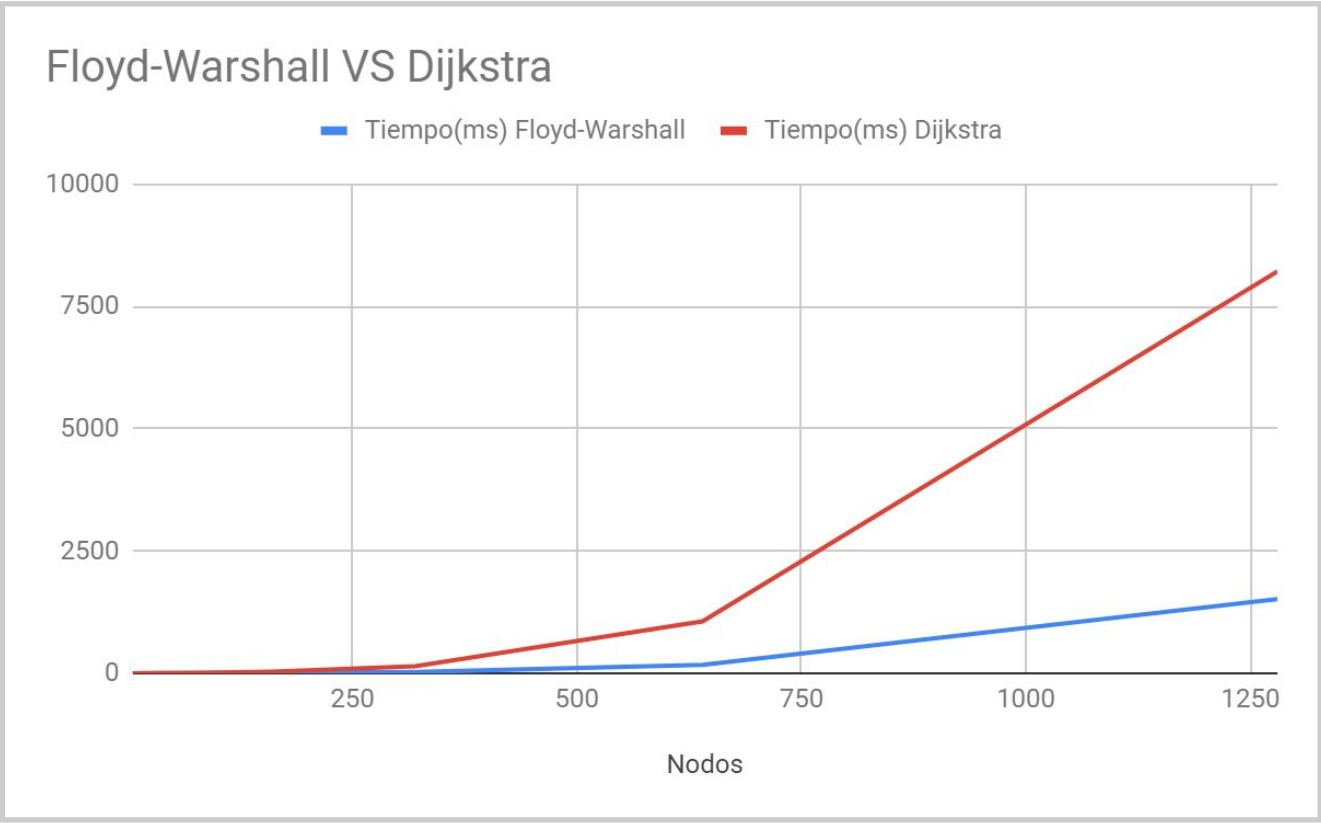
Universidad de La Laguna

Curva teórica



Analisis experimentales

Curva Experimental



Analisis experimentales

Universidad de La Laguna

Conclusiones

- Ventaja de un algoritmo sobre el otro.
- Algoritmos más eficientes.
- El número de aristas influye en la mejora.

Universidad de La Laguna

Referencias

Bibliográficas

Referencias bibliográficas

Shortest path problem, (s. f). En Wikipedia. Recuperado el 24 de Marzo de 2017 de https://en.wikipedia.org/wiki/Shortest_path_problem

Graph theory, (s. f). En Wikipedia. Recuperado el 22 de Marzo de 2017 de https://en.wikipedia.org/wiki/Graph_theory#Computer_science

Dijkstra's algorithm, (s. f). En Wikipedia. Recuperado el 22 de Marzo de 2017 de https://en.wikipedia.org/wiki/Dijkstra%27s_algorithm

Floyd-Warshall algorithm, (s. f). En Wikipedia. Recuperado el 21 de Marzo de 2017 de https://en.wikipedia.org/wiki/Floyd%E2%80%93Warshall_algorithm

MIT OpenCourseWare. (31 de diciembre de 2017). [Video de Youtube]. Recuperado de https://www.youtube.com/watch?v=NzgFUwOaolw

Universidad de La Laguna

Por útimo:

¿Preguntas?

Carlos Troyano Carmona - alu0100822816@ull.edu.es
Miguel Bravo Arvelo - alu0101031538@ull.edu.es