ALGORITMOS AVANZADOS SOBRE GRAFOS

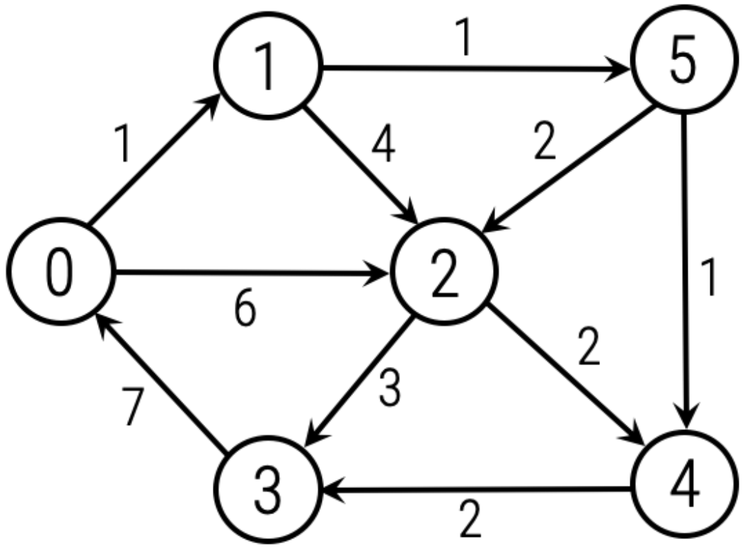
Aplicaciones de Dijkstra y TSP

Problema de las jarras de agua

WIKIPEDIA

$$[8,0,0] \rightarrow [3,5,0] \rightarrow [3,2,3] \rightarrow [6,2,0] \rightarrow [6,0,2] \rightarrow [1,5,2] \rightarrow [1,4,3]$$

Camino mínimo con un número par de aristas



Problema del viajante (TSP)

Encontrar (en un grafo completo) un ciclo Hamiltoniano de coste mínimo

Problema del viajante (TSP)

```
int V; // vértices del grafo completo
vector<vector<int>> dist; // matriz de adyacencia del grafo
vector<vector<int>> memo; // tabla de DP

// devuelve el coste de ir desde pos al origen (el vértice 0)
// pasando por todos los vértices no visitados (con un bit a 0)
int tsp(int pos, int visitados) {
   if (visitados == (1 << V) - 1) // hemos visitado ya todos los vértices
      return dist[pos][0]; // volvemos al origen
   if (memo[pos][visitados] != -1)
      return memo[pos][visitados];

   int res = 1000000000; // INF
   for (int i = 1; i < V; ++i)
      if (!(visitados & (1 << i))) // no hemos visitado vértice i
         res = min(res, dist[pos][i] + tsp(i, visitados | (1 << i)));
   return memo[pos][visitados] = res;
}
```

Problema del viajante (TSP)

En el programa principal:

```
dist.assign(V, vector<int>(V, 0));

// rellenamos la matriz de distancias entre vértices


memo.assign(V, vector<int>(1 << V, -1));
cout << tsp(0, 1) << '\n';
```