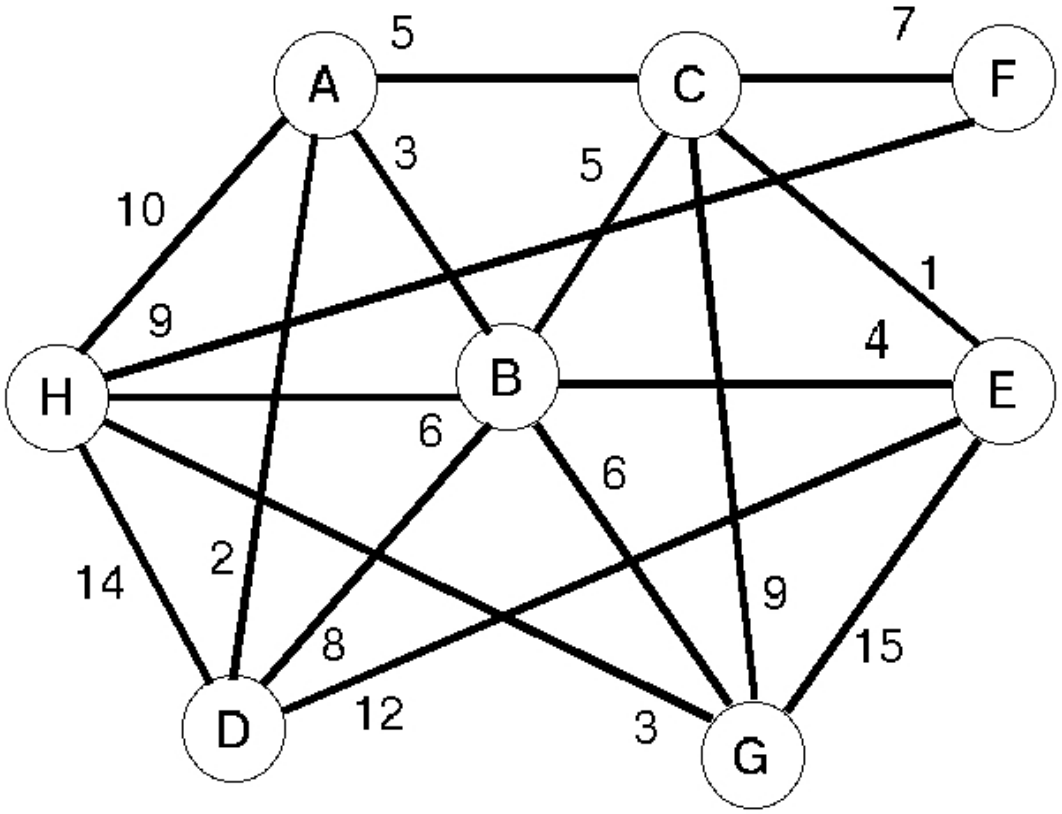
Algoritmo de Dijkstra, parte 1

El problema del camino más corto

Consiste en encontrar un camino entre dos nodos de tal manera que la suma de los pesos de las aristas que lo constituyen sea mínima.



Fuente: http://commons.wikimedia.org/wiki/File:Caminosmascortos.jpg

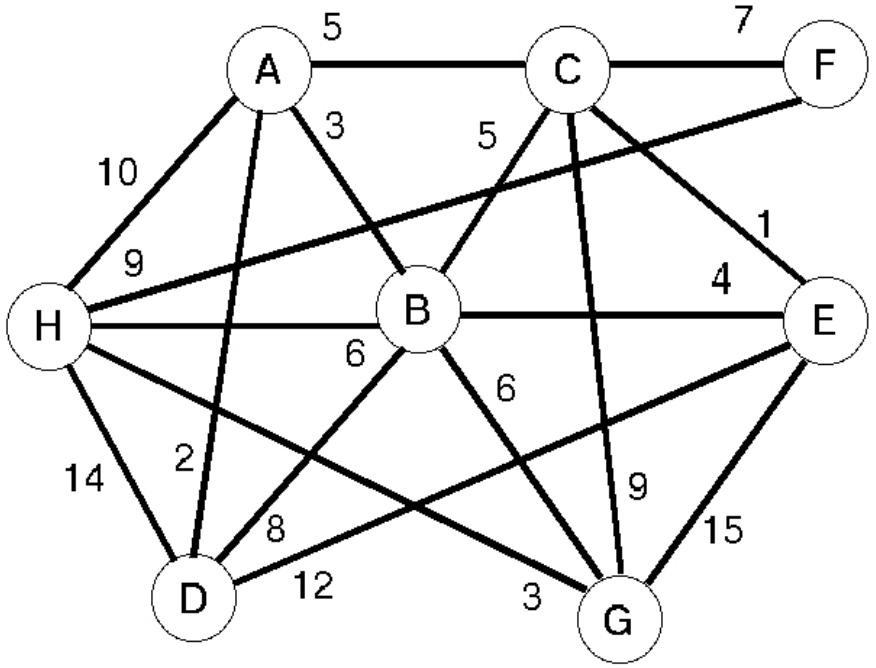
El problema del camino más corto

Entrada: Un grafo dirigido $G = (V, E)$ y un nodo inicial $s \in V$.
Cada arista (u, v) tiene una longitud no negativa l_e.

Salida: Para todo nodo $v \in V$, $L(v)$ = distancia del camino más corto entre s y v.

Si $s = B$:

$L(A)$	3
$L(B)$	0
$L(C)$	5
$L(D)$	5
$L(E)$	4
$L(F)$	12
$L(G)$	6
$L(H)$	6



El problema del camino más corto

¿Por qué no simplemente usar BFS para calcular distancias entre nodos?

Porque dicha aproximación supone que $L(v) = 1$ para todo $v \in V$ y por tanto determina el número de saltos, más que la longitud del camino entre nodos.

Por esta razón es necesario utilizar otro algoritmo, siendo uno de los más populares el de Dijkstra (http://es.wikipedia.org/wiki/Edsger_Dijkstra)

Algoritmo de Dijkstra

```
function Dijkstra(grafo G, nodo a):
   C = {a} //Nodos ya revisados
   a.SPD = 0
   while C ≠ V:
      entre todas las aristas (v, w) Є V con v Є C y w ¬ Є C,
      escoger (v,’ w’) que minimize v.SPD + (v, w).le
      C.add(w’)
      w’.SPD = v’.SPD + (v’, w’).le
```

Algoritmo de Dijkstra