Trabajo Práctico 4 - Ejercicio 5

Materia: Programación III

Carrera: Tecnicatura Universitaria en Desarrollo de Aplicaciones Informáticas

Facultad: Facultad de Ciencias Exactas

Universidad: Universidad Nacional del Centro de la Provincia de Buenos Aires

Autor: Leonardo Magariños

Email: leomagari.os@gmail.com

Fecha de Entrega: 23/05/2018

Enunciado

Desde un cierto conjunto grande de ciudades del interior de una provincia, se desean transportar cereales hasta alguno de los 3 puertos pertenecientes al litoral de la provincia. Se pretende efectuar el transporte total con mínimo costo sabiendo que el flete es más caro cuanto más distancia tiene que recorrer. Dé un algoritmo que resuelva este problema, devolviendo para cada ciudad el camino que debería recorrer hacia el puerto de menor costo.

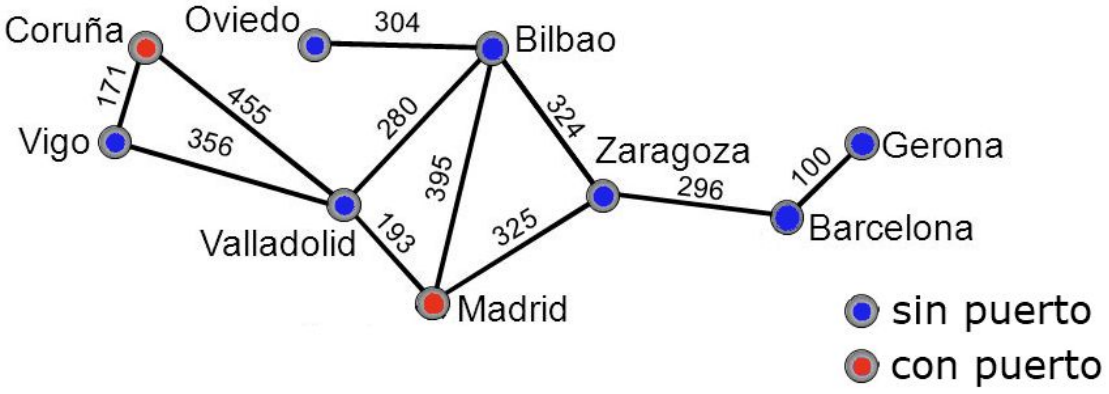
PseudoCódigo

```
function dijkstra (Grafo g, Vertice origen){
    distancia[ ];
    padre[ ];
    visto[ ];
    foreach( g.getVertices() as u ){
       distancia[u] = INFINITO;
       padre[u] = NULL;
       visto[u] = false;
    }
    distancia[origen] = 0;
    Lista candidatos;
    candidatos.add(origen,distancia[origen]);
    while(!candidatos.isEmpty()){
       u = extraerMinimo(candidatos);
       visto[u] = true;
       foreach(u.getAdyacentes() as v){
          if(visto[v]==false){
             if(distancia[v] > distancia[u] + v.pesoTotalA(u)){
                distancia[v] = distancia[u] + v.pesoTotalA(u);
                padre[v] = u;
                candidatos.add(v,distancia[v]);
             }
          }
    }
    caminos=padre;
    return caminos;
}
```

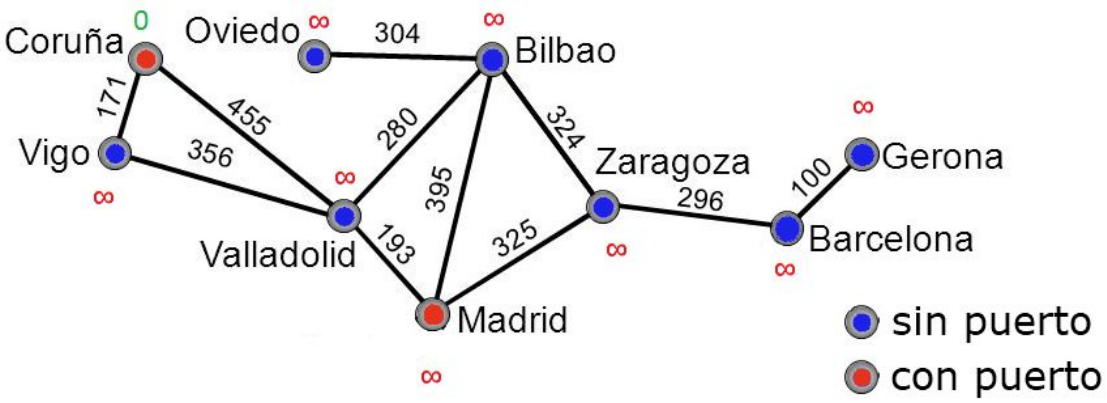
```
function caminoMasCorto(Grafo g){
  ciudadesPuerto=g.getCiudadesPuerto();
  caminos=array();
  foreach ($ciudadesPuerto as $ciudadPuerto) {
    $caminos[]= DIJKSTRA($g,$ciudadPuerto);
  }
  $ciudades=g.getCiudades();//devuelve todas las ciudades menos las
ciudades puerto
  HashMap<ciudad,camino> caminosMasCortos;
  foreach ($ciudades as $ciudad) {
    foreach ($caminos as $camino) {
      if(camino.getVertices().contains($ciudad)){//Factible
          if(camino.calcularRecorrido($ciudad) <
caminosMasCortos.get(ciudad).calcularRecorrido($ciudad))
          {//Seleccionar
               caminosMasCortos.put($ciudad, $camino);
          }
      }
    }
  }
  return caminosMasCortos;//Solucion
}
```

Seguimiento

Se realizará el seguimiento del algoritmo en base al grafo detallado en el siguiente gráfico

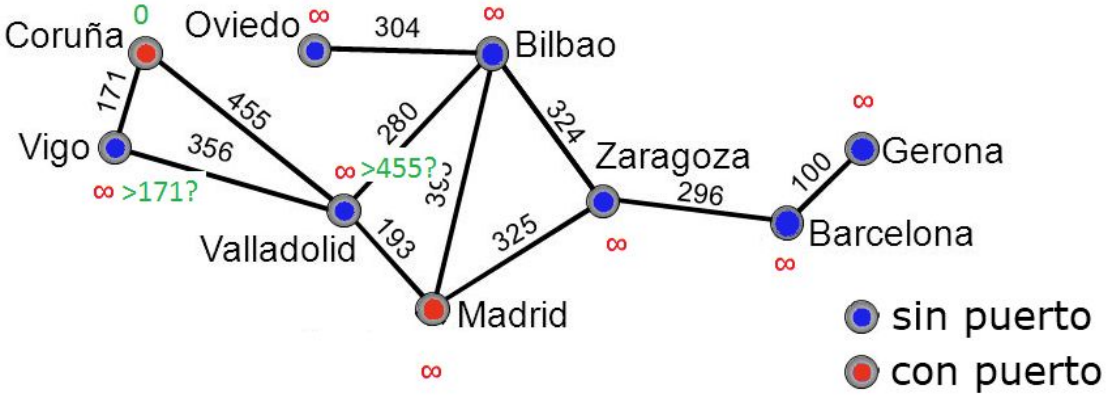


Paso 1



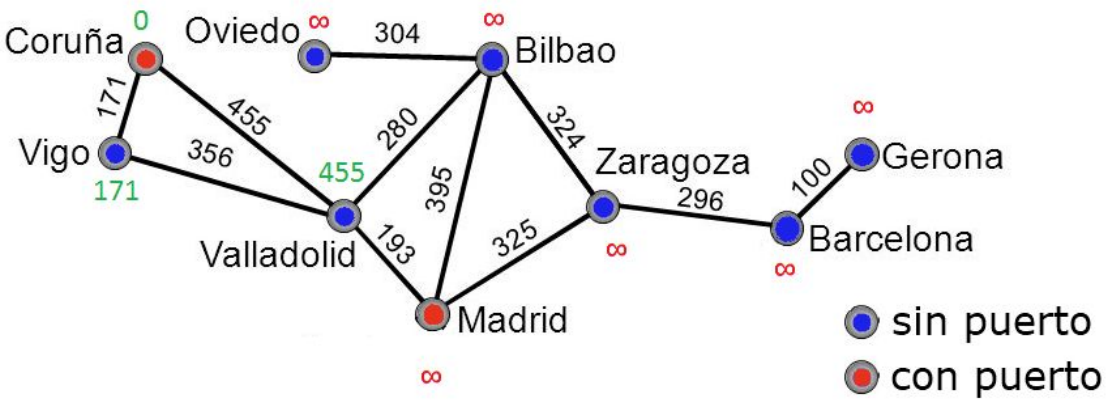
En este paso elegimos la ciudad Portuaria que en el algoritmo llamamos origen, se la inicializa con una distancia de 0 y se la añade al conjunto solución tal que S={Coruña} , también se establece que el peso actual de las demás ciudades será de infinito.

Paso 2-a

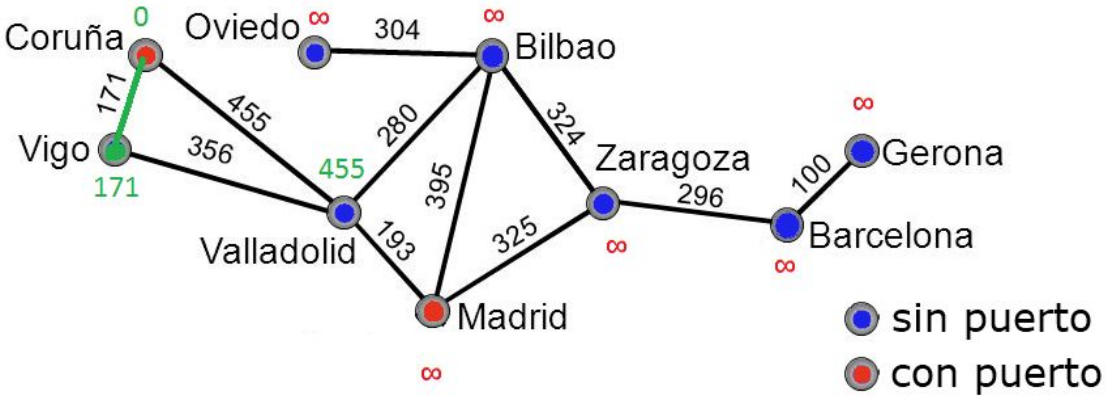


En este paso se actualiza el peso actual de las ciudades que son adyacentes a la ciudad origen, bajo la condición de si el valor que tenían es mayor al peso calculado en este momento. en caso de ser verdadero se actualiza su peso y en el arreglo de vértices “padre” se asocia la ciudad origen como padre de la ciudad que estaba siendo evaluada, también se agrega el vértice a la lista de candidatos.

Paso 2-b

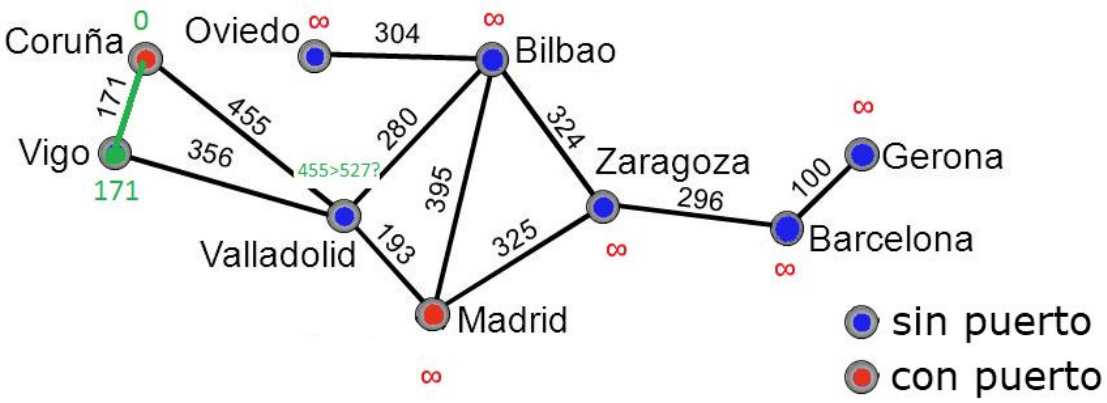


Paso 3



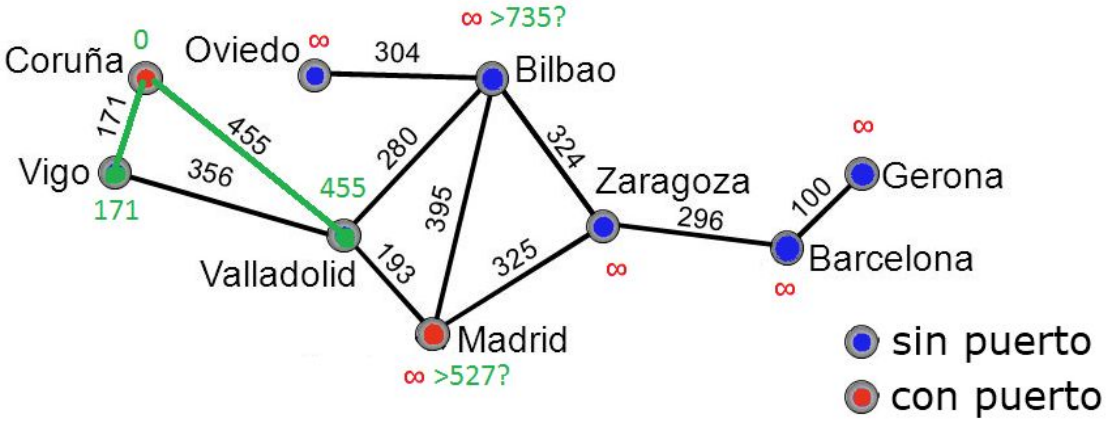
En este paso se elige el vértice con menor peso de los que se agregaron a la lista de Candidatos para continuar con el algoritmo.

Paso 4



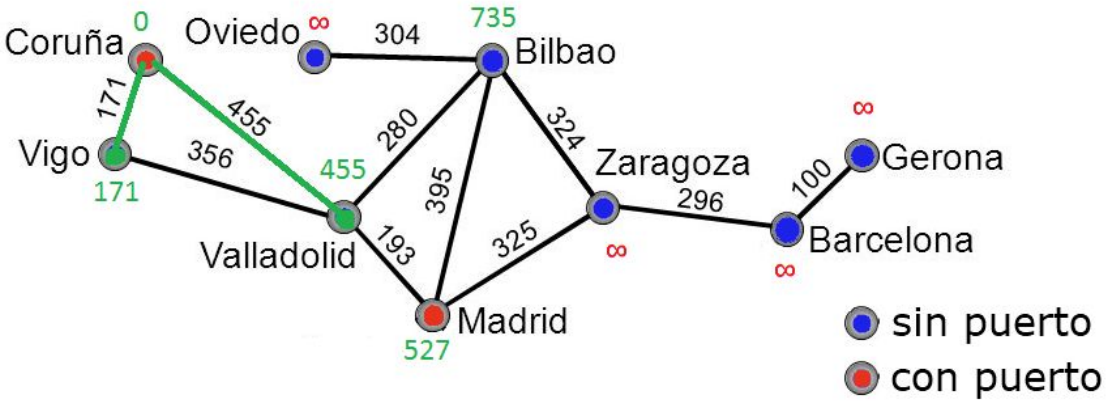
En este paso se realiza lo mismo que en el paso 2, se trata de actualizar el peso del vértice adyacente al actual, y como la condición no se cumple se avanza por el siguiente candidato.

Paso 5

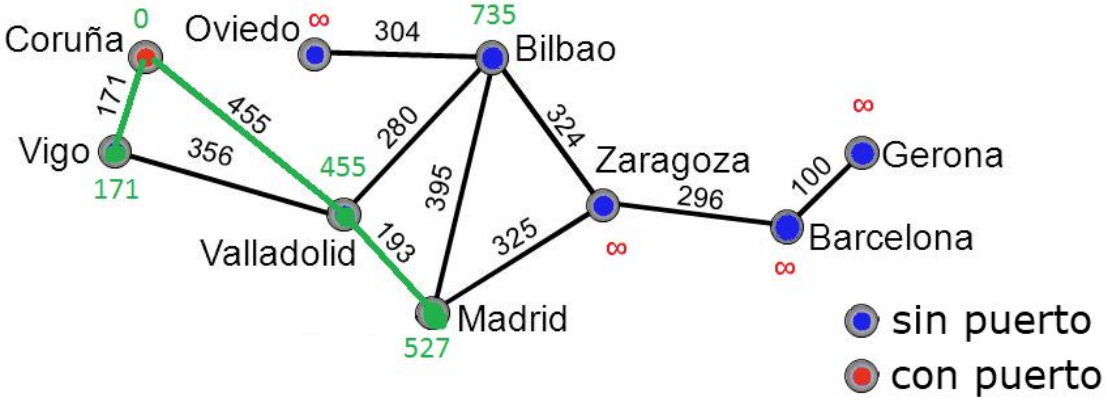


En este paso se continúa con la actualización de pesos y se cambio de vértice al siguiente en la lista de candidatos.

Paso 5-b

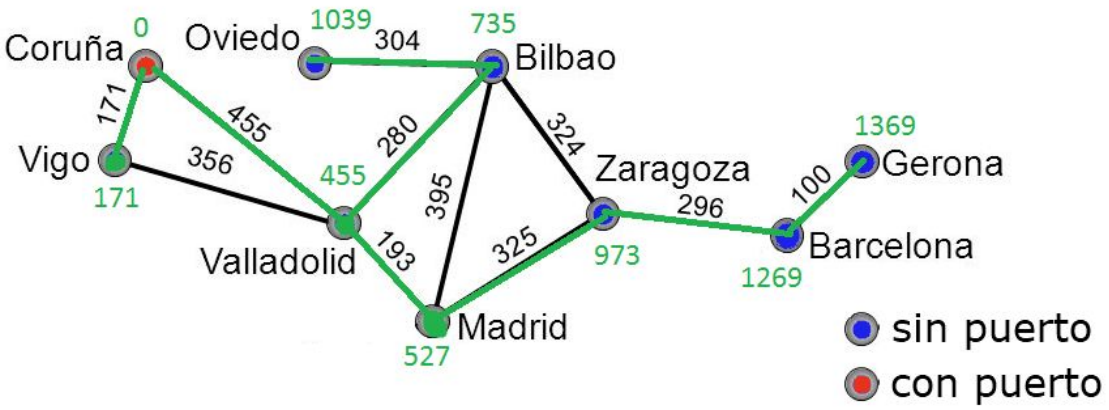


Paso 6



En este paso se continúa con el siguiente en la lista de candidatos.
Se prosigue con el algoritmo hasta llegar al paso final

Paso Final



En este paso se observa cómo se fue eligiendo el camino más corto entre el vértice origen y los demás vértices, se muestran los pesos actualizados en cada nodo.
Este algoritmo devuelve un HashMap siendo la clave el vértice y el valor el padre del vértice clave.