RepartidorDeBebidas.pdf

Anónimo

Estructuras de Datos no Lineales

2º Grado en Ingeniería Informática

Escuela Superior de Ingeniería
Universidad de Cádiz

REPARTIDOR DE BEBIDAS (2022)

//Un repartidor de una empresa de distribucion de bebidas
// tiene que visitar a todos sus clientes cada dias.
//Pero, al comentar su jornada de trabajo, no conoce que cantidad de bebidas
//tiene que servir cada cliente, por lo que no puede planificar una ruta
//optima para visitarlos a todos. Por tanto, nuestro repartidor decide llevar
// a cabo la siguiente estrategia:

// -El camión parte del almacen con la máxima carga permitida rumbo
// a su cliente más próximo
//- El repartidor descarga las cajas de bebidas que le pide el cliente.
// si no tiene suficientes cajas en el camion, le entrega todas las que tiene.
// Este cliente terminara de ser servido en algun otro momento a lo largo del dia
// cuando la estratefia de reparto vuelva a llevar al repartidor hasta él
//- Después de servir a un cliente:
// - Si quedan bebidas en el camion, el repartidor consulta su sistema de navegacion basado en el GPS
// para conocer la ruta que le lleva hasta su cliente más proximo pendiente de ser servido
// - Si no quedan bebidas en el camion, vuelve al almacén por el camino más corto y otra vez carga
// el camión completamente
// - Despues de cargar el camión, el repartidor consulta su sistema de navegacion y se va por el camino mas corto
// a visitar al cliente pendiente de ser servido más procimo

// Implementa un subprograma que calcule y devuelva, la distancia total recorrida en un dia por nuestro repartidor
// a partir de lo siguiente :
// - Grafo representado mediante la matriz de costes con las distancias de los caminos directos entre
// los clientes y entre ellos y la central
// - Capacidad máxima del camión (cantidad de cajas de bebidas)
// - Asumiremos que existe una funcion int Pedido() que devuelve el numero de cajas que quedan por servir
// al cliente en el que se encuentra el repartidor

//NOTA: Es absolutamente necesario definir todos los tipos de datos implicados en la resolucion del problema ,
// así como los prototipos de las operaciones utilizadas de los TADS conocidos y tambien los prototipos de los
// algoritmos de grafo utilizados de los estudiados en las asignaturas

```
float calcularDistanciaTotal(const GrafoP<tCoste>& distanciasDirectas, const int
capacidadMaximaCamion) {

   matriz <vertices> P;
   matriz <tCoste> m = Floyd(distanciasDirectas, P);
   int cajasDisponiblesCamion = capacidadMaximaCamion;
   int numTotalParadas = distanciasDirectas.numVert();
   vertice paradaActual = 0, destino; // asumimos que 0 es el almacen
   tCoste valorMinimo = INF; //coste a la ciudad mas cercana
   vector<bool> ciudadesAbastecidas(distanciasDirectas.numVert(), false);
   int i;
   float distanciaTotalRecorrida = 0 ;

   while (cajasDisponiblesCamion > 0) {
      i = 1;
      //while (paradaActual != 0 && i != numTotalParadas ){
      while (i != numTotalParadas) {
         if (m[paradaActual][i] < valorMinimo) {
            if (paradaActual != i && !ciudadesAbastecidas[i]) {
               valorMinimo = m[paradaActual][i];
               destino = i;
               i++;
            }

            if (destino.Pedido() <= cajasDisponiblesCamion) {
               cajasDisponiblesCamion = cajasDisponiblesCamion - destino.Pedido();
               ciudadesAbastecidas[destino] = true;
            } else {
               destino.Pedido() = destino.Pedido() - cajasDisponiblesCamion;
               cajasDisponiblesCamion = 0;
            }
            distanciaTotalRecorrida += m[paradaActual][destino];

            if (cajasDisponiblesCamion == 0) {
               distanciaTotalRecorrida += m[paradaActual][0];
               paradaActual = 0;
               cajasDisponiblesCamion = capacidadMaximaCamion;

            } else paradaActual = i;
         }
      }
   }
```

```
    return distanciaTotalRecorrida;
}
```