UNIVERSIDAD EAFIT

ESTRUCTURA DATOS Y ALGORITMOS 2

DOMICILIOS

DOCENTE:
MAURICIO TORO

ALUMNOS:
JUAN GONZALO QUIROZ CADAVID
ALEJANDRO DIAZ CANO

Posibles soluciones al problema:

a) El algoritmo de Christofides

b) 2-opt (Pairwise) 5% más aproximado que el algoritmo de Christofides

c) k-opt (Lin – Kernighan), funciona como 2-opt pero en un tour dado eliminará K arcos diferentes. Usando búsqueda local, podemos encontrar algoritmos como:

d) Markov chain es un algoritmo que se usa para grandes número de ciudades (700-800) y es extremadamente cercana a la ruta más óptima.

e) Ant colony Optimization es un algoritmo que genera soluciones próximas a las más óptimas, simula una colonia de hormigas, las zuleta en el mapa y ellas encuentran un camino entre la comida y la colonia, a medida que esta encuentra una ruta deja unas pheromones, representado en número como inversamente proporcional a la longitud del camino, entre más rápido sea el camino más depositara.

Finalmente decidimos usar el Algoritmo A* en conjunto con el Algoritmo de Help

ESTRUCTURA DE DATOS

La estructura de datos fue imaginada por Juan Gonzalo Quiroz,

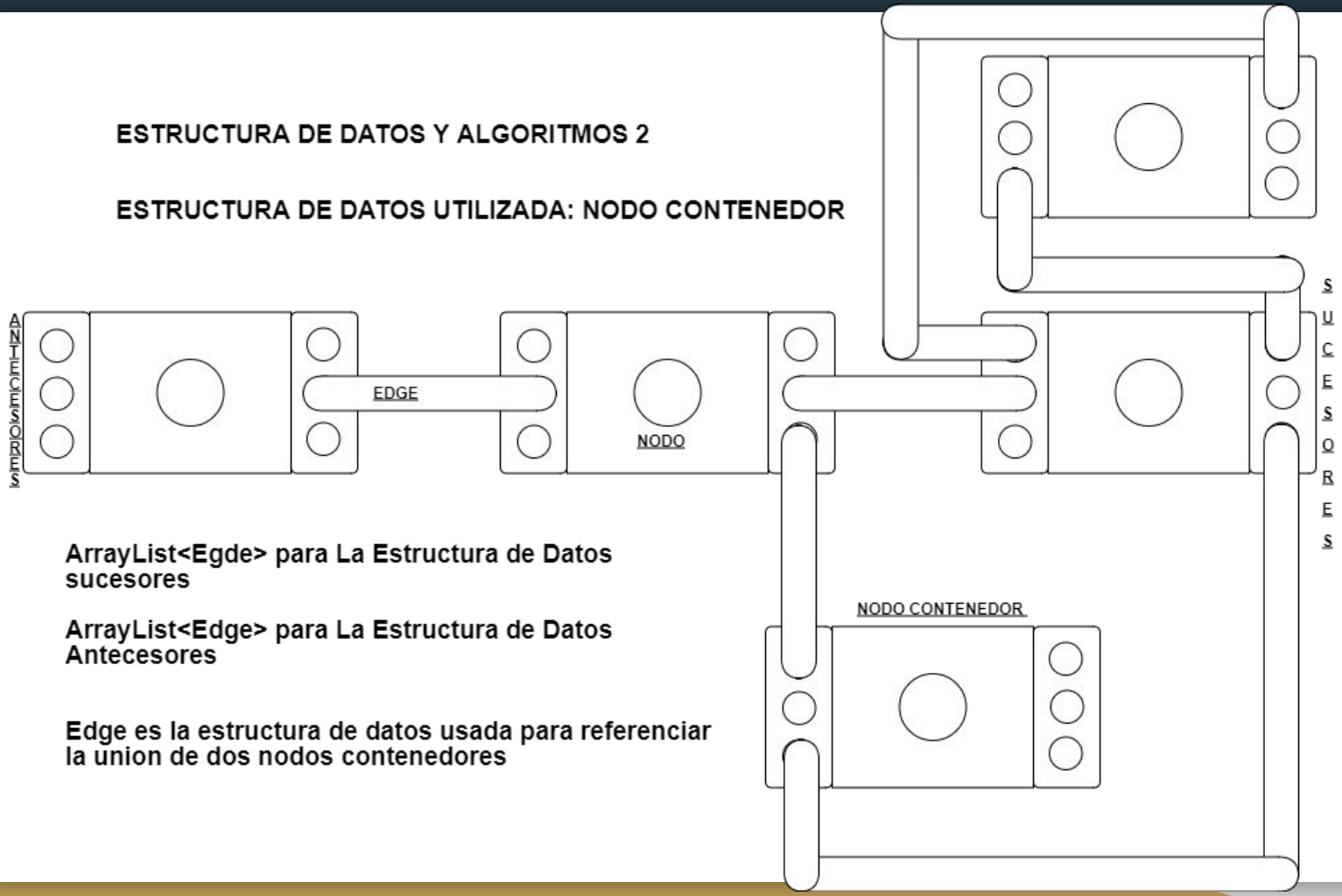
consiste en que cada nodo tiene para sí mismo dos arrayList en donde

están almacenados todos sus antecesores y sucesores, conectado entre si quienes tienen comunicacion y llevandolos a lo mismo con otro nodo de la misma característica, en nuestro codigo es llamado nodeContainer. creando una especie de red con redes interiores,

buscando así crear un algoritmo más eficaz para recorrer más fácilmente

ESTRUCTURA DE DATOS Y ALGORITMOS 2

ESTRUCTURA DE DATOS UTILIZADA: NODO CONTENEDOR



ArrayList<Egde> para La Estructura de Datos sucesores

ArrayList<Edge> para La Estructura de Datos Antecesores

Edge es la estructura de datos usada para referenciar la union de dos nodos contenedores

Complejidad del Algoritmo:

SUB PROBLEMA	COMPLEJIDAD
Agregar Nodo	O(n)
Nodos Sucesores	O(n)
Nodos Antecesores	O(n)
Organizar Por Nombre	O(n)
Conectar Nodos Contenedores	O(n)
Algoritmo de Busqueda A*	O(v+e) , O(v^2)
Algoritmo Help Carp	O(2^n * n^2)
Complejidad total	O(v^2 + 2^n * n^2)

¿Por que el algoritmo diseñado es bueno?

El Algoritmo diseñado es bueno porque:

Buscamos de una forma más eficiente el camino de un nodo con todos los demás nodos, mediante sus respectivos ArrayList que contiene antecesores y sucesores.

Tenemos un nodo inicial y nodo objetivo(s) miramos sus sucesores y antecesores del nodo inicial con el fin de encontrar el nodo objetivo.

obteniendo la distancia más corta posible comparando nodo con nodo, creando un nuevo arreglo con los visitados y vamos guardando

y sumando su peso para seguir comparando, cuando llegamos al nodo ya tenemos los visitados y con el Algoritmo de Help carp creamos la matriz con la obtenemos costos mínimos contenedor con contenedor de nodos creando una optimización. La forma de búsqueda es por anchura con el Algoritmo A*