<!--Universidad Nacional Del Comahue-->

Sistema de Mudanzas Compartidas

<Por="Matias Tartaglia"/>

}

Consigna{

Se desea desarrollar un sistema para una empresa que realiza mudanzas compartidas entre localidades de Argentina.

El principal objetivo del sistema será permitir consultar sobre qué camino tomar para cubrir viajes entre una ciudad de origen A y una ciudad destino B de forma eficiente.

Requisitos

- Eficiencia
- Ejecucion por separado
- Modularizacion
- Estructuras genericas
- Carga Inicial
- Archivo log

}

Estructuras {

DICCIONARIO(ARBOL AVL)

Almacena informacion de las ciudades usando como clave el codigo postal de las ciudades(en tipo Comparable)y una instancia de clase Ciudad correspondiente al codigo postal.

Funciona como arbol auto-balanceado, por lo tanto siempre va a estar equilibrado y la complejidad de busqueda, o eficiencia se mantiene en orden O(log n).

El arbol se conforma por un conjunto de nodos enlazados de la clase NodoAVLDicc, que guardan la clave Comparable, un dato(en este caso una instancia de ciudad) y los enlaces de hijo izquierdo y derecho

¿Porqué Arbol AVL?

- **Con cada inserción la estructura garantiza que siempre este balanceada lo que deja una eficiencia de orden O(log n).**
- **Esta eficiencia garantiza una búsqueda eficiente de ciudades, que se ordenan por un codigo postal unico, por lo que 2 ciudades no van a tener la misma clave.**
- **La forma generica de la estructura de arbol AVL permite esta eficiencia ademas de poder almacenar la instancia de clase ciudad,con todos los datos completos, en sus nodos.**

}

Estructuras {

GESTOR SOLICITUDES

La clase gestor solicitudes utiliza una instancia de la clase HashMap de Java para almacenar una Lista con solicitudes asociadas a un par unico de ciudad de Origen con ciudad de Destino.

Utiliza como clave para el hash una instancia de la clase Pair que tiene los codigos postales de origen y destino.

Como un par de ciudades puede tener mas de una solicitud se utiliza una lista para almacenar las solicitudes a esas 2 ciudades de manera un poco mas eficiente, sin tener solicitudes que se pierden,se repiten o no se cumplen.

¿Porqué HashMap?

- **La forma del HashMap garantiza una busqueda eficiente de pares de ciudades sin permitir que haya elementos vacios,o repetidos.**
- **El utilizar una lista para alamecenar las solicitudes que hay entre 2 ciudades evita que no queden solicitudes sin estar asociadas a un par de ciudades, si no que ese par van a tener todas las solicitudes entre las 2 ciudades.**
- **Ademas el usar pares de ciudades lo hace idoneo para lo que pide el ejercicio, a la hora de verificar un viaje por ejemplo se busca el par de manera eficiente y se devuelve un listado con las solicitudes de ese par**

}

Estructuras {

GRAFO ETIQUETADO

Guarda informacion sobre las rutas entre algunas ciudades.La carga inicial forma un grafo conexo,es decir, que de un nodo x se pueda llegar a cualquier nodo y.

Almacena los códigos postales de las ciudades y las distancias entre ellas en las etiquetas de los arcos.

Esta implementado usando Listas de adyacencia, usando los dos tipos de nodos, NodoVert(Nodo Vertice),que guarda el codigo postal, el siguiente vertice y el primer adyacente, y NodoAdy(Nodo adyacente) que guarda el vertice de tipo NodoVert y el siguiente adyacente.

¿Porqué Grafo Etiquetado?

- **Este tipo de estructura almacena unicamente el codigo postal de las ciudades y las distancias entre algunas de ellas.**
- **Una vez realizada la carga inicial, el grafo se convierte en un grafo conexo lo que permite hacer cualquier recorrido entre las distintas ciudades.**
- **Los metodos permiten hacer listas de caminos, ya sea caminos mas directos o caminos mas cortos en lo referente a distancia.**
- **Ademas esta implementado el metodo que permite obtener todos los caminos posibles, pasando por una ciudad intermedia lo que devuelve una lista de listas con caminos posibles para realizar**

}

Estructuras {

MAPEO A MUCHOS(IMPLEMENTACION DE HASH ABIERTO)

Almacena informacion de los clientes utilizando como clave el tipo de documento y el numero de documento.

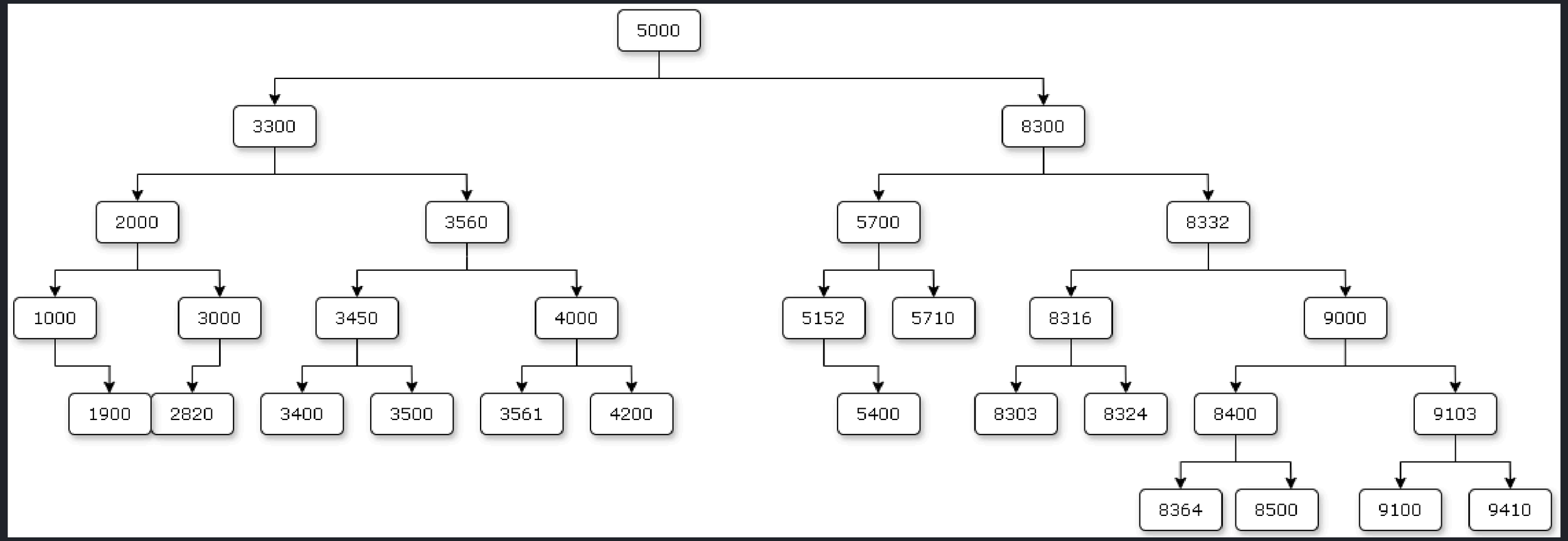
Fue implementado utilizando la estrategia de HashAbierto, por lo que guarda un arreglo de NodosHashMapeo que a su vez guardan el dominio(TipoDocumento y NumeroDocumento), el rango(la instancia de Cliente)y el enlace al siguiente nodo.

¿Porqué Hash Abierto?

- **Al tratarse de un Hash, utilizando la clave garantiza una busqueda eficiente de orden constante u O(1).**
- **La implementacion permite que se le asigne u tamaño adecuado a la carga, asi que si se conoce la escala de la carga permite que el rendimiento no se deteriore.**

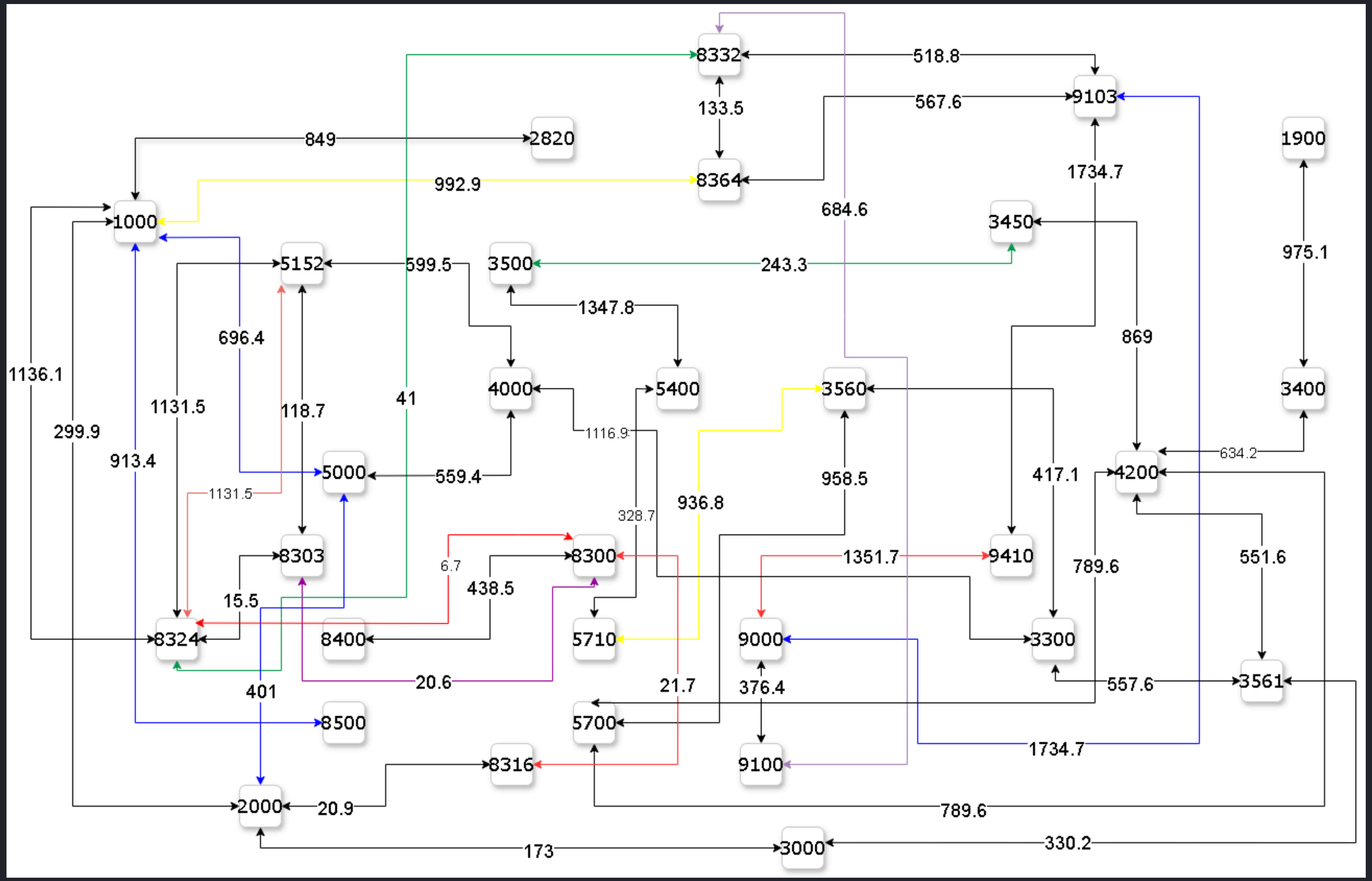
}

Arbol AVL de la carga Inicial{



}

Mapa de rutas de la carga Inicial{



}

El Código {

Todo el codigo y documentacion relacionada con este trabajo practico se puede encontrar en este enlace

}

<!--Facultad de Informatica-->

Gracias {

<Por="Matias Tartaglia"/>

}