



Final Project

Miguel Angel Montoya Zaragoza
Luis Eduardo Vargas Victoria
Ricardo Palma Mendoza



The purpose

Elaborate a time-efficient algorithm that can keep track of the relation between an user's belongings and the ones from another user. The project will simulate the interaction of a user and his accounts as a member of a certain bank.



Functionality

The project will simulate the interaction of a user and his accounts as a member of a certain bank.



Depth-first Search Algorithm

Looking for the furthest node traverse the nodes of the graph with this.



Breadth-First Search

Looking for the closest node traverse the nodes of the graph with this.



Edmonds Algorithm

Get the spanning arborescence (minimum spanning tree) of the graph, use a unique relation to traverse all the nodes.



Hashing

Simple for the clients' id and the accounts' id.
Necessary for insertion in both hash tables. The hash tables implemented will use linear probing.




Heapsort

To sort the accounts that belong to a client. Keeping the account that's been used the most as heap.



Queue

When a movement happens from an account into another, that movement is enqueued, so that a relation between both accounts involved is stored.



Action	Average Case	Worst Case
Create New User	$\theta(1)$	$O(U)$
Search User	$\theta(1)$	$O(U)$
Delete User	$\theta(1)$	$O(U)$
Create New Account	$\theta(1)$	$O(U+A+\log(A))$
Search Account	$\theta(1)$	$O(A)$
Delete Account	$\theta(1 + \log(A))$	$O(U + A + \log(A))$
Make Transaction	$\theta(1)$	$O(U + 2A + \log(A))$
Search Transaction(DFS)	$\theta(1 + A + T)$	$O(2A + T)$
Search Transaction(BFS)	$\theta(1 + A + T)$	$O(2A + T)$
Edmonds's	$\theta(A + A^2 + T)$	$O(A + A^2 + T)$

NOTE:

A = Accounts U = Users T = Transactions (T has a constant value of 15 as maximum)