EXISTING SYSTEM

RANKED VOTING METHOD

Cloud computing provides computing resources on demand. To assist the users, for selection of a best service provider as per user's requirement, it is necessary to create a solution.

STEP1:

Enter one ballot per line. A ballot consists of ranking numbers 1 2 3 4 ... in a row separated by a comma.

STEP2:

A ballot has the same amount of ranking numbers as there are candidates. The numbers are in the same order as the candidates, the first number is the rank of the first candidate on this ballot and so on.

STEP3:

Incomplete ballots can be created by using 0 as a placeholder or leaving the position empty like this 1, 0, 2, , 3.

STEP4:

Instant runoff voting uses ranked choice ballots to simulate a traditional runoff in a single round of voting. Voters rank candidates in order of preference. They typically are given the option to rank as many or as few providers as they wish. Indicating support for a lesser choice never counts your higher choices.

STEP5:

Every voter has one vote. That vote is counted initially for a voter's first choice.

STEP6:

If there are more than two providers who receive votes, the last-place provider with the fewest votes is eliminated. More than one provider can be eliminated simultaneously if their combined vote is less than the total of any other remaining candidate.

STEP7:

Ballots counting for the eliminated candidate are added to the totals of the candidate ranked next on each ballot.

This process of eliminating last-place candidates and adding ballots cast for those candidates to the totals of the next-ranked choice on that ballot continues until two candidates remain.

STEP8:

The candidate with the majority of votes in this final round is declared the winner

STEP9:

Voters rank the candidates and those first choice rankings are tallied as in Option One.

STEP10:

The count is stopped at any point that a candidate as a majority of votes cast for continuing candidates - either by having a majority of first choice rankings or a majority of total votes in any round of counting.

PROGRAM:

import java.util.ArrayList;

import java.util.List;

import java.util.Iterator;

import java.util.Random;

import java.util.Scanner;

import java.util.Set;

import java.util.Map;

import java.util.TreeSet;

import java.util.TreeMap;

import java.io.\*;

public class InstantRunoffProc {

private static boolean debugOn = true;

private static Random generator = new Random();

public static Set<String> getInitialCandidates(List<List<String>> ballots) {

Set<String> candidateSet = new TreeSet<String>();

for (List<String> ballot : ballots)

if (ballot.size() > 0)

candidateSet.add(ballot.get(0));

return candidateSet;

}

public static Map<String, Integer> countVotes(Set<String> candidates,

List<List<String>> ballots) {

Map<String, Integer> voteTally = new TreeMap<String, Integer>();

for (String candidate : candidates)

voteTally.put(candidate, 0);

for (List<String> ballot : ballots) {

String topChoice = getTopChoice(ballot, candidates);

if (topChoice != null) {

int currentTally = voteTally.get(topChoice);

voteTally.put(topChoice, currentTally + 1);

}

}

return voteTally;

}

public static String getTopChoice(List<String> ballot, Set<String> candidates) {

Iterator<String> iter = ballot.iterator();

while (iter.hasNext()) {

String candidate = iter.next();

if (candidates.contains(candidate))

return candidate;

}

return null;

}

public static ArrayList<String> getLosers(Set<String> candidates,

Map<String, Integer> voteTally) {

ArrayList<String> loserList = new ArrayList<String>();

int minTally = Integer.MAX\_VALUE;

for (String candidate : voteTally.keySet()) {

int candidateTally = voteTally.get(candidate);

if (candidateTally < minTally) {

loserList.clear();

loserList.add(candidate);

minTally = candidateTally;

}

else if (candidateTally == minTally)

loserList.add(candidate);

}

return loserList;

}

public static String pickRandomItem(ArrayList<String> list) {

return list.get(generator.nextInt(list.size()));

}

public static String runInstantRunoffElection(List<List<String>> ballots) {

Set<String> candidates = getInitialCandidates(ballots);

while (candidates.size() > 1) {

Map<String, Integer> voteTally = countVotes(candidates, ballots);

String loser = pickRandomItem(getLosers(candidates, voteTally));

candidates.remove(loser);

if (debugOn) {

System.out.println("Vote tally:\n" + voteTally);

System.out.println("Loser: " + loser);

}

}

if (candidates.size() > 0)

return candidates.iterator().next();

else

return null;

}

public static void main(String[] args) {

List<List<String>> ballots = new ArrayList<List<String>>();

List<String> ballot;

System.getProperty("File.seperator");

String ballotFileName ="D:\\b.txt";

Scanner ballotFile = null;

try {

ballotFile = new Scanner(new File(ballotFileName));

}

catch (FileNotFoundException e) {

System.err.println("No such file: " + ballotFileName);

System.exit(1);

}

while (ballotFile.hasNext()) {

String line = ballotFile.nextLine();

ballot = new ArrayList<String>();

Scanner inLine = new Scanner(line);

while (inLine.hasNext()) {

String candidate = inLine.next();

ballot.add(candidate);

}

ballots.add(ballot);

}

String winner = runInstantRunoffElection(ballots);

if (winner != null)

System.out.println("The winner is: " +

winner);

else

System.out.println("No valid votes cast");

}

}

TEXT FILE:

hp,ms,amazon

hp

ms

hp,amazon

amazon,hp

ms,amazon

hp,amazon

ms

amazon

hp

OUTPUT:

GRAPH: