Code Docs

A 'Point' is a data structure in the form (v_k, i_k), where v is voltage and I is current.

The ArrayList of points that represents the input data from the hardware (energy meter) is initialized as 'originalTraj'.

A 'Cell' is a data structure that contains the following information:

'leftPoint' : voltage

'rightPoint' : current

'binP' : binary value indicating whether or not that cell is traversed by originalTraj

'posx' : x-axis position of the cell

'posy' : y-axis position of the cell

The 2N x 2N grid described in step 5 is initialized as 'grid', as a 2D ArrayList.

'halfCycle' is an ArrayList<Cell> and refers to the data points from the first 'zero-crossing' point in originalTraj. I am temporarily adding a dummy cell to the front of halfCycle, and am calling it 'startingCell'. In practice, this needs to be the first 'zero-crossing' point.

'this.setupVal()' takes as arguments an ArrayList<Point>, and an int. The ArrayList<Point> must be taken from the hardware data, and this functionality needs to be implemented. The int refers to the n value, which is also taken from the hardware. Information about the n value can be found in 'VI-TrajToBinaryAlgo.pdf'

'this.setupVal()' also calls 'this.initValues()', and 'this.initializeGrid()', while initializing 'n' to the argument nVal, , 'originalTraj' to traj, and temporarily setting halfCycle to a new ArrayList<Cell>

'this.initValues()' finds maxV, minV, maxI, and minI, by looping through 'originalTraj'. At the end, it calls 'this.initVar()'

'this.initVar()' finds values for Vo. Io, deltaV, and deltaI, according to instructions from the research paper.

‘this.initializeGrid()’ is called by this.setupVal(). The function assigns positional values on the basis of instructions from the research paper, and also sets the binaryPosition value to 0 for every cell. At the end, it calls ‘this.findWinners()’.

‘this.findWinners()’ loops through halfCycle, and sends each cell to ‘this.setWinner()’ in order to continuously update the winner cell. The winner cell is stored as a variable, ‘winner’.

The loop described in step7 of ‘VI-TrajToBinaryAlgo.pdf’ is coded in ‘this.setWinner(Cell, int)’. The int argument is passed as a ‘winner’ cell should only be passed to the next function, ‘this.searchNeighbours’ if the int > 1. This is due to step 8 of ‘VI-TrajToBinaryAlgo.pdf’, where it specifies ‘remaining data points’.

‘this.searchNeighbours()’ first checks if the Cell being passed in is on the border of the grid or not. If it is not in the border, it checks if any of the eight adjacent cells have been traversed or not.

Questions/Clarifications:

Step 9 has not yet been implemented in code, as I am not sure what the predefined number is.

What to do when there is more than one winner per cell, for step 7.

Original instructions for the creation of the algorithm are available in Section 3 of “VI-Algo.pdf”, in /Misc