Automata Theory: The Firing Squad Problem Alternation

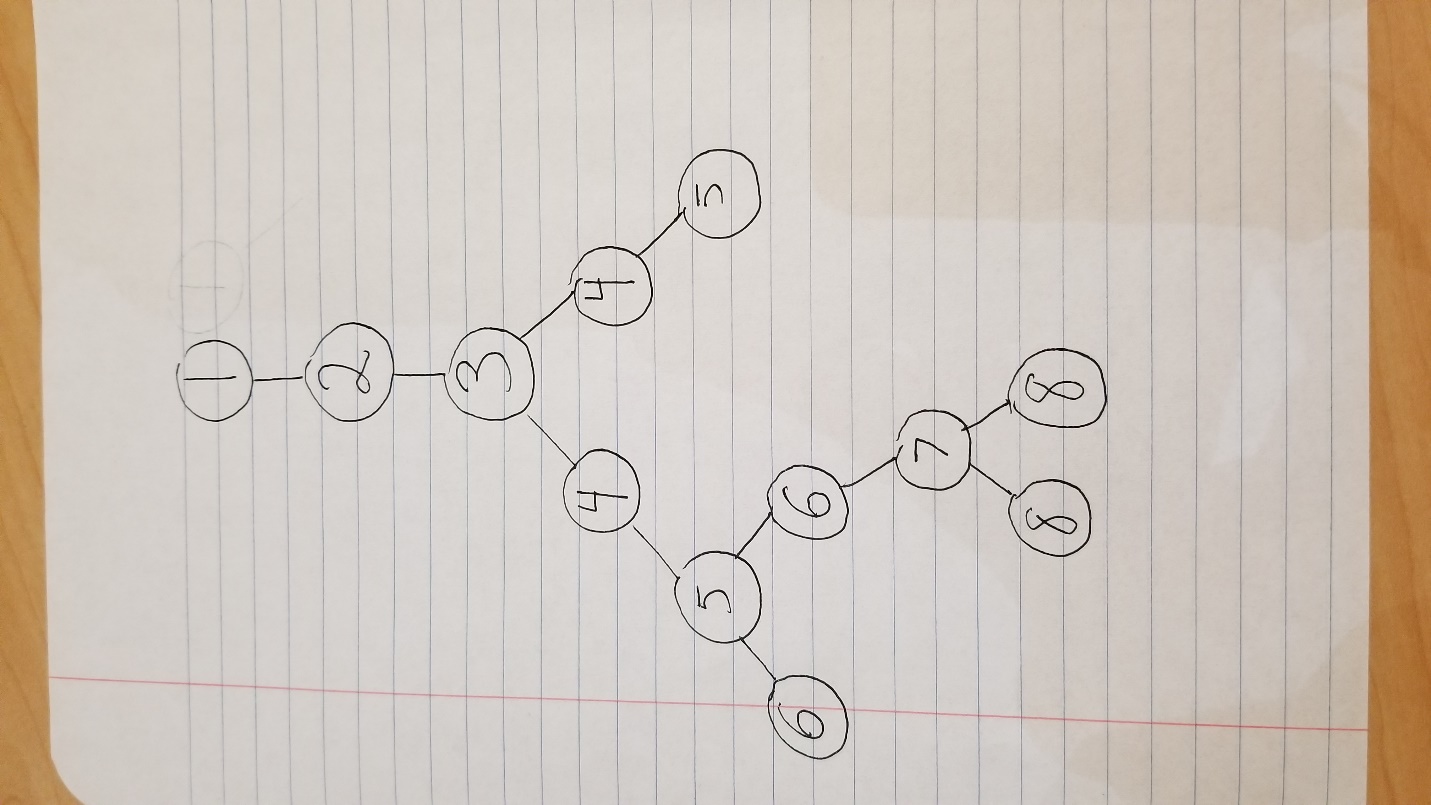
Claim: The firing squad can split off into branches and there exists an algorithm to fire all nodes at the same time.

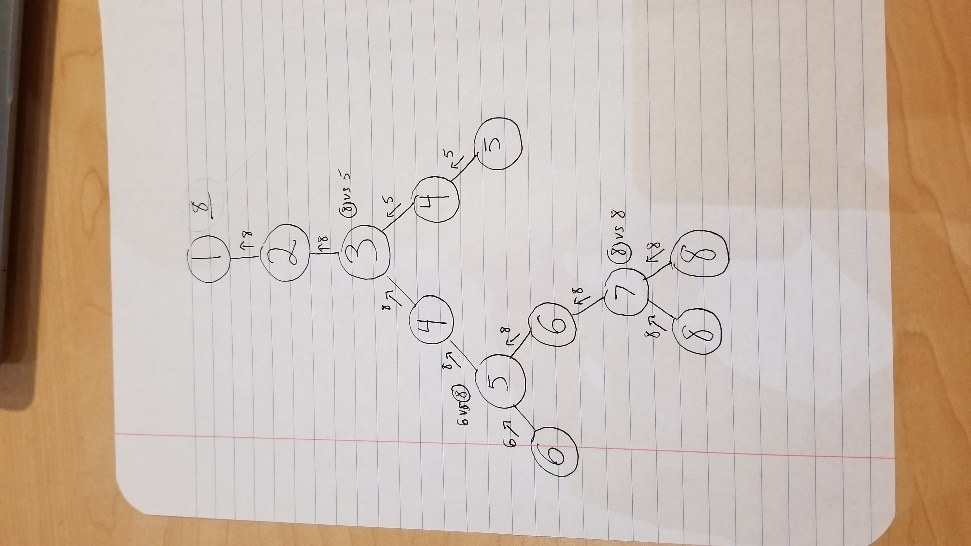
Whether the soldiers are in a straight line, a circle, or a line that branches off into sub-branches, there will always be a solution to fire all nodes simultaneously. Given there are no branches whose links intertwine, no matter how the line looks whether it is curved, wavy, criss-crossed, etc., it can always be redrawn as a straight line. It has been proven that it is possible to fire all soldiers at the same time regardless of where the general is. Now imagine there is a larger hierarchy of soldiers that information needs to be passed down to. A General passes down information to a Sergeant and the Sergeant has two or more battalions filled with soldiers they’re in charge of creating branches. Think of this as a Tree with the general as the head. There exists an algorithm in which all soldiers can fire at the same time.

In order to have all soldiers fire at the same time, the following rules have to be set in place:

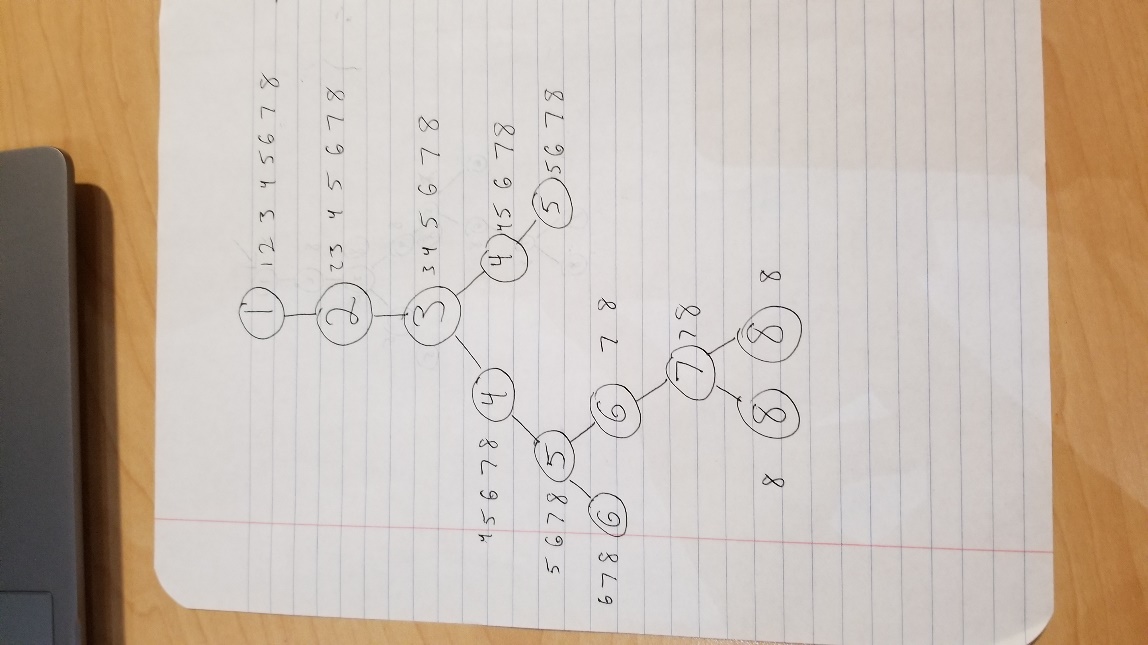
1. All soldiers keep time at the same pace.
2. Each soldier can only communicate with those directly linked to them
3. Every soldier knows what layer of branch they are in (represented by a number - duplicates are allowed since children of a node are the same layer meaning they share a number)
4. Every commanding soldier knows how many subordinates they have (each parent node knows how many children it has)
5. Commanding officers can receive and compare information from their immediate subordinates (parent nodes can receive and store information from its children and compare it).

Before anything can happen, the chain of command (the tree) needs to be established (created). Start at the head and work down the tree assigning a number for each layer.



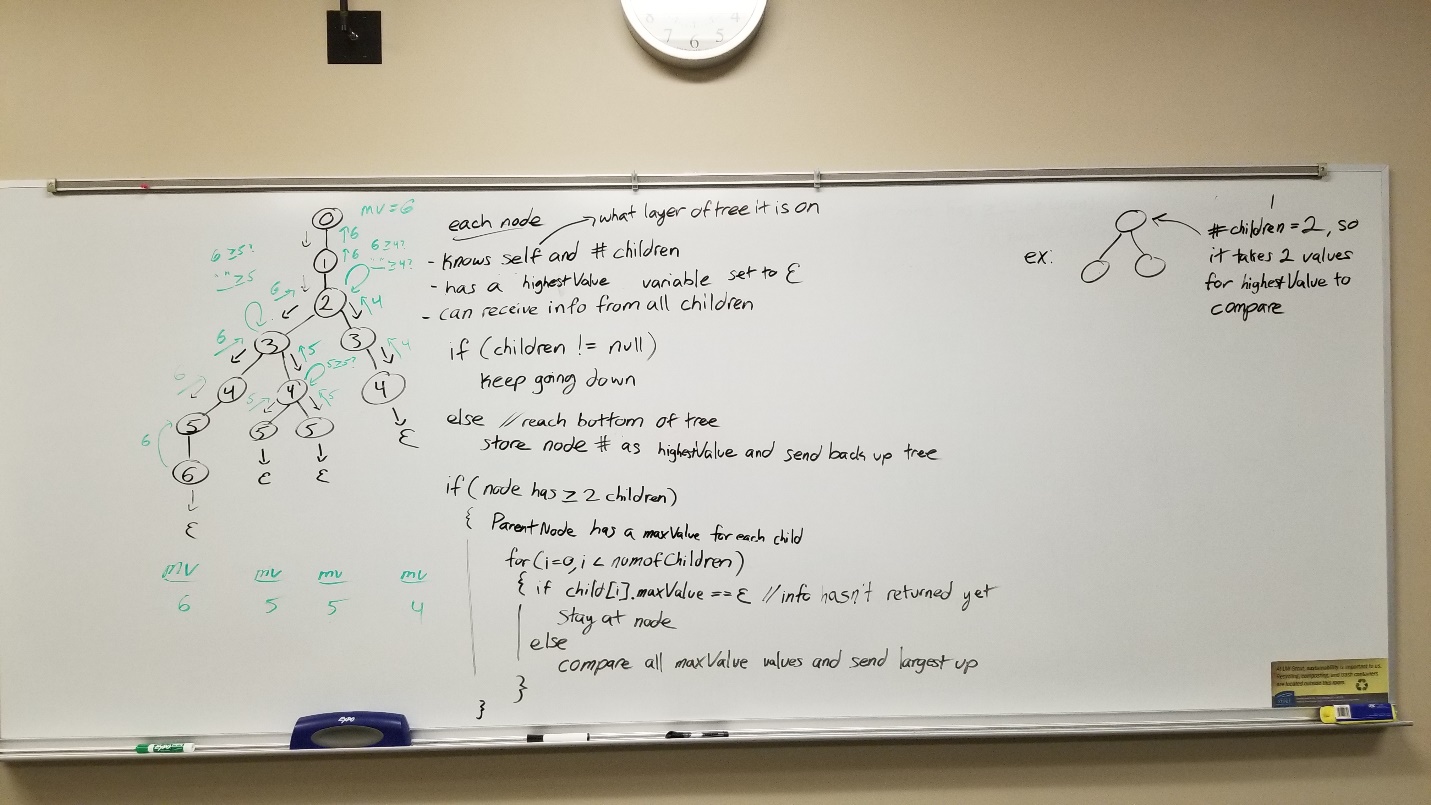
Once the tree has been established, we can begin. Starting at the head, begin traversing down the tree. If a node has a child, the parent node tells all its children to continue going downwards simultaneously. E.g referring to the picture above, after traversing from node 2 -> 3, 3 sends a signal to both children 4L and 4R simultaneously. Since this scenario referring to soldiers, two people getting info from one person simultaneously is allowed. This continues for each branch until it reaches the bottom of each branch. Once the bottom of a branch is reached, that bottom node passes its value to its parent and begins to traverse back up the tree. If a parent node only has the one child, it passes the value up. If a parent node has two or more children, it holds on to the number its child sent to it and waits until all the other children pass in their values. Once all children pass the values to the parent, the parent compares all values passed into it and sends up the largest value. This process continues until the head is reached.

At this point, the node that is furthest away is known by the General (the head). This is the point where having all soldiers keeping time at the same rate and knowing what layer they are in is crucial. The General now has to pass this information back down the chain of command so all soldiers are informed. However, when passing down the max value, along with receiving the max value, the soldiers begin to count starting from their number and then fires once they count to said max value. Since every soldier is counting at the same rate, by the time the bottom of the tree is reached everyone will reach the max value and all soldiers will fire simultaneously.



The photos above were relatively simple to give a general understanding. The following photos include more detail and general pseudocode as a proof of concept.

This photo is a more detailed visualization of what is happening with paths going down in black arrows and returning paths in green.



\*correction: in the if statement, remove the “else” and put the “compare all maxValue values and send largest up” outside the for loop. “Stay at node” also implies checks on each iteration to see if that node still doesn’t have a maxValue and then continues once it does.

This photo better demonstrates what happens after the head receives the max value. Each row represents what information is currently at each node. Each column represents traversing down the tree. E.g. at node 0 in column 1, since it knows the max value to count to, it begins counting starting at itself, but nodes 1-6 don’t have any info yet. Going to the second column, node 0 continues counting while node 1 begins counting, but since it was stated the nodes start counting at themselves, both nodes 0 and 1 are at the same count. This continues all the way down through all nodes. At the end, all nodes reach the max value at the same time and fire.

