Coursework 1: Forest Fires and Reactive Agents

Agent Design

The agent I have designed to complete the forest fire simulation is a reactive agent that uses the subsumption architecture, which means the agent is made up of task – accomplishing behaviours. Regarding the example at hand, one of the tasks the agent is asked to do is put out a fire if there is a fire detected in the region of 8 patches. The task in this example is putting out the fire ("*put-out-fire*") and the accomplishing behaviour is putting the fire, another sensor in this example if the need for the agent to check if it has any water before it can put out a fire, which can be seen within the line of code in the 'execute behaviour' class: *if have-water and detect-fire [put-out-fire].* This fits the subsumption model as there is no symbolic reasoning or representations.

As this agent is designed with subsumption architecture a key aspect is the hierarchy of the behaviours in the model. I have designed my code so that the lowest ranked behaviour is the check for an obstacle and in turn to avoid the obstacle (*"if detect-obstacle [avoid-obstacle stop]"*), I have put this as the top priority as I believe this is the behaviour that will be called the most, given no parameters are changed from start up. This means that the code will first be checking for obstacles and would require less processing, plus this is a task that is key for the agent to put out all the fires.

The next tasks involve the water that the agent is carrying, so the next lowest in my hierarchy is the refilling of water when at the base. This means that the agent will make sure that it has any water before it starts searching for fires, saving time within the model as it will not get to a fire and not be able to extinguish it ("*if at-base and need-water [service-unit]*"). The next is the sensor that will start the movement towards the base if there is no water ("*if need-water [move-towards-base]"*) so the behaviour for servicing the unit can take place when reaching the base ("*if at-base and need-water [service-unit]*"), which is placed at the third highest point in the hierarchy.

The behaviour for putting out the fire is called when the sensors for 'have water' 'detect fire' are satisfied, this is placed in between the check for needing water and if the unit is at the base and has water start moving ("*if have-water and at-base [move-randomly]*") as the task is more important and will be called more often than moving from the base after servicing but, as mentioned above, it is not possible to put out a fire without water. (*"if have-water and detect-fire [put-out-fire]*")

Experimental Evaluation

To evaluate the performance of my model I edited both the initial water carried by the agents and the number of agents in the model. Within these parameters I am exploring the possibilities of having less agents but carrying more water, as it could be more economical for whoever this model is intended for to have less agents in a real-world situation. This will show efficiency of a low number of agents when loaded with water. To look at another possibility, I have decided to increase the amount of agents but decrease the amount of water as this may be more efficient way to put out the fire as this will decrease the amount of times an agent goes back to the station to refill as it would be more likely another agent will have found all of the fires before this point.

Both of those changes in parameters are more focused on finding the most efficient combination in a perfect world, the next two tests are more of a practical look at how the model will run given the restriction of a parameter. The first is looking at how the system would work given there is an average number of agents at the disposal of the fire extinguishing team but only having the ability to carry a small amount of water, this will simulate a situation that could be like a real-world problem if only smaller agents are available to help. The model will show how effective they could be and if they need larger agents. The last test is looking at a reduced number of agents available but with an average initial water value. This is like a real-word problem as it could be possible that some agents are being used for another task, so we will need to see what the minimum threshold of agents would be for us to successfully put out all the fires with an average amount of water. This will show us when we need to start looking at reassigning agents from other tasks.

Results:

The results in Table 1 correspond with the first test I outlined in the experimental evaluation section above, from this table you can see that when there were less agents available to the model they still struggle to extinguish a substantial amount of fires. The test showed that even if all of the model had the maximum amount of water (50) available to it 174 trees still died, this could be because when the agent is carrying more water it is slower meaning it cannot get to the fires at the edge of the map before it kills the trees.

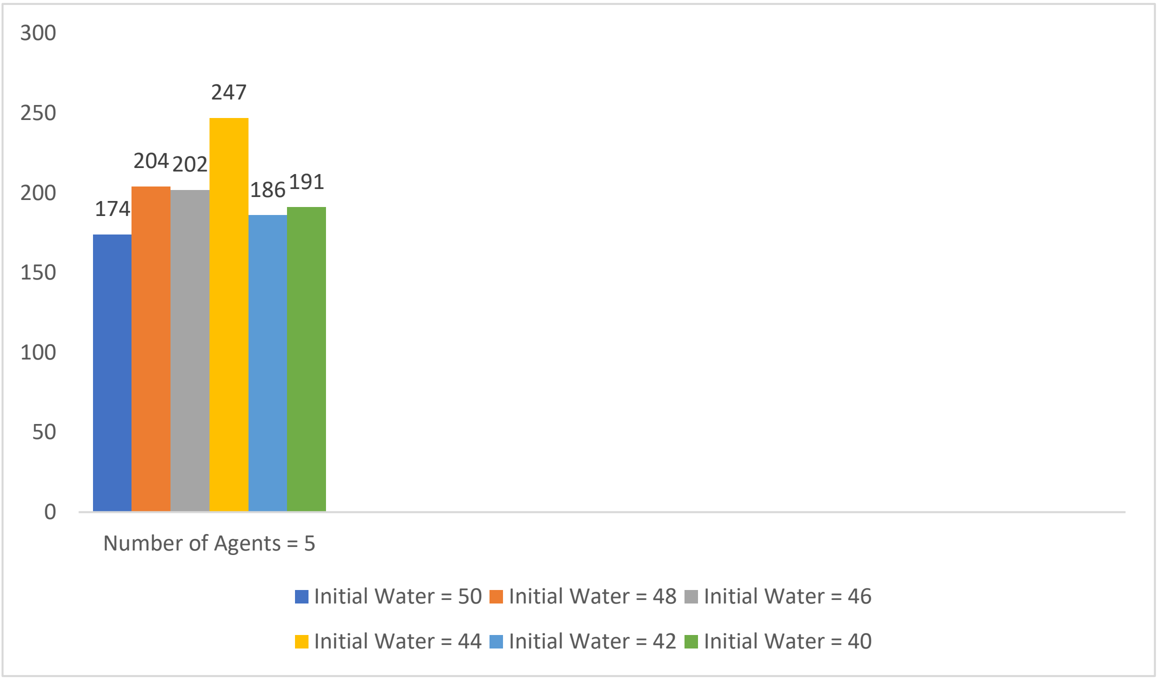

Table 1 - Number of Dead Trees charted for 5 agents with full access to water

The results in table 2 correspond with the second test I planned in the above section, this table shows that when there are many agents available the initial water value can be as low as 5 to extinguish the large majority of the fires, some numbers below 10 even extinguished all of the fires. When the number reached the high teens and into the twenties the number of dead trees increased as the agents were having the same problem as in table 1.

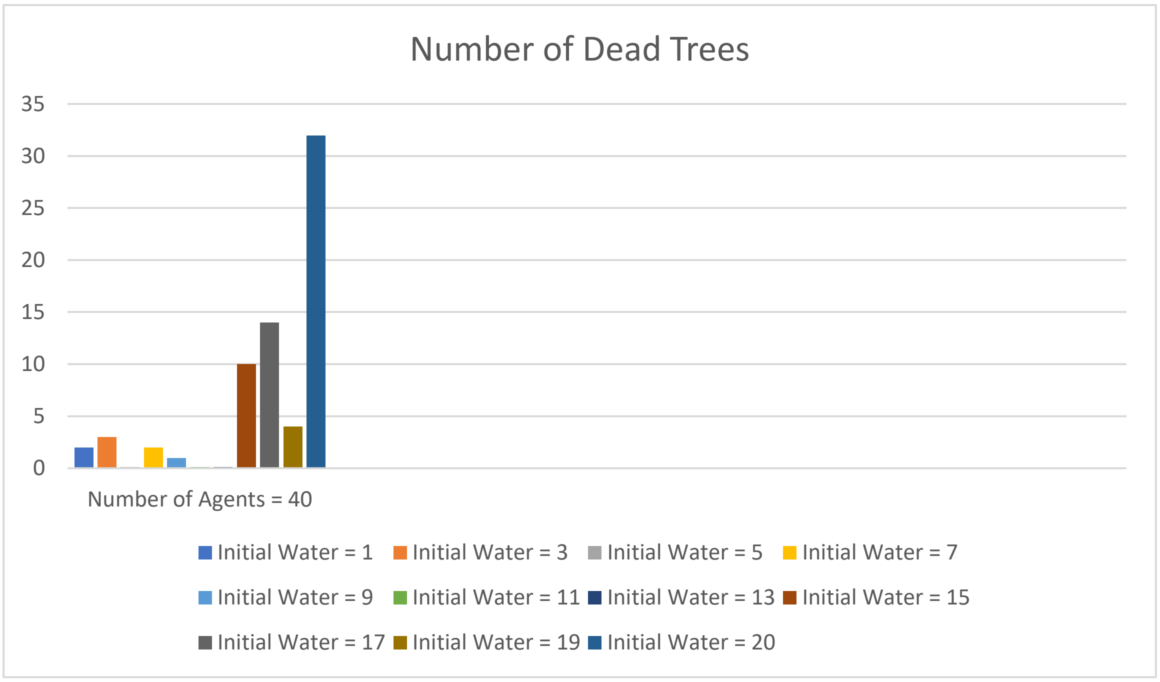

Table 2 - Number of dead trees captured with 40 agents and limited water available

The results in table 3 show the testing the average number of agents with limited water capacity, you can see from the table that the amount of trees saved is much higher than the amount of trees that died this could be because the agents didn't need to travel back to the base to refuel and were not slowed by carrying excessive amounts of water.

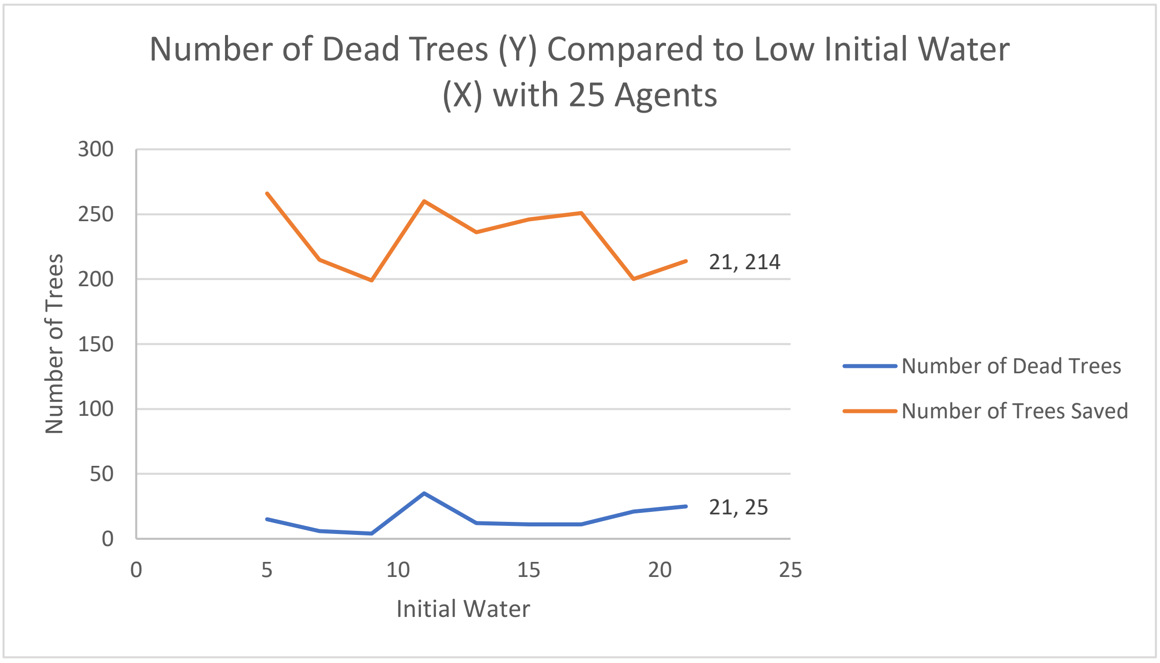

Table 3- Number of dead trees captured with 25 agents with low initial water

The results in table 4 show the effect of a medium amount of initial water when tested with 5 agents, the results show that the lower the amount of initial water the more effective a small number of agents will be, this adds more proof for above hypothesis and suggests that 5 agents will struggle to extinguish all the fires in an area this size.

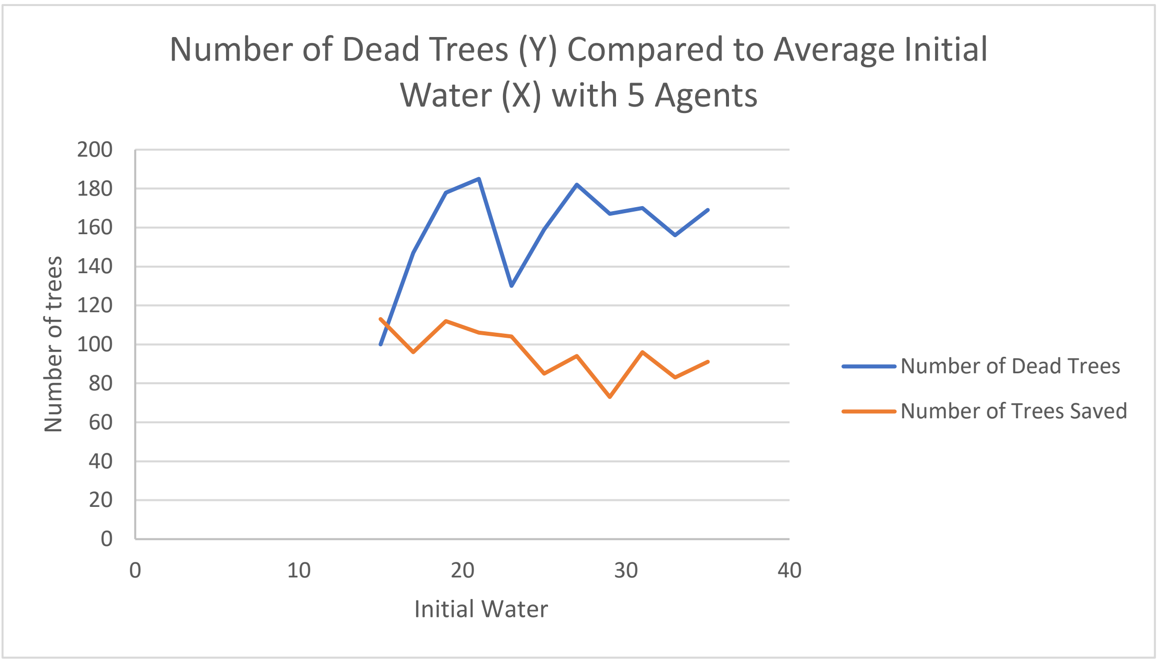

Table 4- Number of dead trees captured with 5 agents with average initial water

In conclusion, it seems the most effective combination of the two variables is to have a large number of agents with a low initial amount of water, this allows the agents to go into the world and extinguish a small amount of fires each but as there is enough agents doing the same the majority of the trees get saved. This is also more efficient when there are smaller numbers of agents as the agents are more effective when they are not slowed by carrying large amounts of water. Some limitations of my results include the large jump between 5, 25 and 40 agents this makes it hard to compare how the same water value a differing number of agents. Another limitation is that as the agents move randomly within the map the same values of agents and water may produce different results each time, this means it is difficult to rely on the results gathered in my tests.

Improvements to Current System

One improvement that could be made to the system would be the addition of an effector that will be able to extinguish a certain amount of fires within the area. This would work by extinguishing however many fires that are detected in the 8 patches surrounding the agent, this number can be defined in the control panel with a slider, similar to initial water. This would increase efficiency of the agent as it would be able to extinguish multiple fires in one action, potentially putting out a whole connected line of a fire before it spreads any further. This will also be making the agent more intelligent as we would be reducing the amount of random movement across the map by giving the agent a goal at more points in the model as water will be used up quicker so the agent will head back to the base more often.

Another improvement of the agent would be to ask the agent to start making its way back to the base when its water value is below a certain amount (not 0), this would be implemented as a slider that is directly linked to the initial water value, just being a certain percent of the initial water value. This would increase the efficiency of the system as the agent will start towards the base with some water left meaning it will be able to extinguish any fires that it encounters on the way. This increases efficiency as the agent will spend less time not being able to extinguish fires. As the current model does not see strings of trees on fire it will again reduce the amount of random movement in the model.

A final improvement would be adding the ability to check if there is a string of fires attached to the current fire an agent is extinguishing, this would increase efficiency as it would stop the agent from randomly moving away from an area which still has fires near it. An extension of this improvement would be being able to communicate with other agents if there is an area with a fire in allowing the other agents to know if the area is worth exploring or not. Implementation could be achieved by leaving markers on each move if it is returning to the base without any water. This would mean if another agent saw the markers left it would follow them into the area where there is potentially a fire. This increases efficiency of the model by reducing the number of random movements and adding another state where it is heading towards a goal.

Advantages & Disadvantages of Reactive Agent

One advantage of using a reactive agent for the forest fire simulation is that it gives a cheap and robust view of certain situations that could occur if a forest fire was to take place, allowing the modeller to see how current agents would fair in extreme situations e.g. limiting number of agents available to use. Another advantage of using a reactive system is that they are simple to set up meaning that it is easy to add another type of test, expanding the scope of the model easily, giving the modeller a broader overview of a potential forest fire.

One disadvantage of using a reactive agent is that the agent cannot plan before it reaches a situation – difficult to make an agent learn - meaning it cannot check to see if an area has fires in it before it (or another agent in my improved model) has been to the area before, this is wasting time of the agent by exploring areas that don't have any fires in them. By extension of this, another disadvantage is that it that all decisions are made locally, in this example it cannot foresee if a fire will spread further than another by being connected to more trees, if it had this ability it would be able to stop more fires started decreasing the number of trees that die in turn.