AMS Coursework 2

FIRE EXTINGUISHING AGENT

Ali Azam | BSC Computer Science | 1528624

Introduction

The report will discuss the strategy used for implementation, challenges faced, learning curves, experimentation and combination of multiple agents. The report will mainly cover the following parts:

1. Extended Agent
    a. Description and Justification of the design
    b. Description of implementation
2. Experiments
3. Results
4. Advantages of Hybrid Design

Extended Agent

1.a **Description and Justification of the proposed protocol:**

The extended agent still uses the BDI architecture and extends it to add more features such as communication and further beliefs about the world. The agent design is as follows:

The role of the ground units to put out fire and be hybrid agents which act both reactively and proactively, whereas the role of the scouters is to search for fires and inform all the agents about the fire, the design of the agents have not been altered, their roles remain the same as they were, only the communication, beliefs and intentions are added to increase their efficiency.

In a nutshell, the design of the cooperation was kept simple and intuitive to make it extendible and efficient. Most of the things were kept the way they were and only minimalist change was done to increase the efficiency of the program.

The idea is very simple, the scouters should inform all the ground units about the fire and the ground units should still believe there is a fire at a certain coordinate, but not take any action until instructed to do so by the scouter. How does the scouter make the decision on whom to instruct to eliminate the fire? The answer is simple, it chooses the best one based

the distance. The question becomes, how does the scouter know about the distance of all ground units to itself? That's where FIPA communication kicks in and solves the problem.

The communication works as follows:

Scouter: Inform all ground units about the fire.

Ground Units: Calculate the distance from itself to the fire, create a reply message and add the distance as the content of the message and send the reply

Scouter: For every incoming message from the ground unit, check if that's the minimum distance its seen in the content of the message, if yes then update the minimum seen and record the message, if no then ignore it as it already has closest agent who is best fit to extinguish the fire.

Scouter: Once iterated over all messages in the incoming queue, create a reply message and add the contents of the message which will be the fire locations and mark it as fire-locations-to-put-out and send the reply

Ground Unit: The specific ground unit will receive a response and it will add fire location to its beliefs which will then turn into its intention because of the label fire locations to put out. (Further explained below in 1.b)

The motivation of this solutions comes from various day to day processes such as Job application process where the company X advertises their job everyone, then applicants apply for the job from all over the world and X prioritizes applicants who are the closest since they can be the fastest to join. Also, keeping in mind the Hollywood Principle, [http://wiki.c2.com/?HollywoodPrinciple] "*Don't call us, we'll call you*".

It was hard to decide at the earlier stage of the design process on whether the ground agents should be the one to allocate themselves the job and let others pick other jobs,

however this wasn't a feasible solution as if the ground agent which picks it is surrounded by other agents then it will be stuck there and the tree will die.

1.b Description of implementation:

The above description of the protocol gives a brief overview of the implementation and each part of the implementation will be discussed here in detail explaining the decisions made and justifying them.

The first part of the code which was modified is adding the "move-randomly" to the ground agent as the tests showed that when the agents were moving randomly, the trees on fire were put out much faster than when "move-randomly" was not in the agent.

One of the important design decisions made here is that move-randomly was called within the units behavior and not added as an intention, the reason for this is because the agent in my opinion should not have the intention of moving randomly but move randomly when it has nothing better to do, this is more like humans, we wonder around not really looking for something but just wondering around without an intention. (In this case, the intention of "find-target-fire" will be there). With the design difference aside, the behavior of the agent will remain the same in either way.

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;; THE AGENTS ;;;;;;;
;;; unit behaviour consists of two layers. The reactive an pro-active. The reacitive layer
;;; is responsible for extinguishing fire when it is detected and for reloading the agent with water,
;;; if it is necessary. The Proactive layer is repsonsible for receiving messages setting targets to extinguish etc.

to units-behaviour
  if reactive-behaviour-unit [stop] ;; Just to make sure that only one action gets fired.
  collect-msg-update-intentions ;; Read through all the messages that were received, then update the intentions
  execute-intentions  ;; Try to realise intentions
  move-randomly ;; if all intentions are execeuted then move randomly [1]

end
```

Figure 1 Showing the addition of calling move-randomly to make ground units move randomly

The scouter behavior at the start remains the same, it moves around to look for fire and once it has found a fire, it informs all the ground units about the fire and the content of the message is from a function **fire-location-s** (this will be addressed later in the report). The ground units get the message of the fire and they add it to their beliefs as they used to, and they calculate the distance from the location in the message (content of the message) and from their coordinates, with the closest distance, the agent creates a reply message (FIPA library function which automatically adds the sender and receiver) and adds the content to be the distance and sends the message. [Figure 2]

```
;; Collect messages looks at all the messages, and may change the agent's beliefs about the world
;; The beliefs may have changed and so we should update intentions
to collect-msg-update-intentions ;[2] modified to exchange messages
let msg 0            ;; initialise the message
let performative 0   ;; initialise the performative

while [not empty? incoming-queue]
  [ ;; iterate over the message queue
   set msg get-message  ; get the message from the queue and set to to the variable message
   set performative get-performative msg   ; set the performative to the performative of the message, a local variable

   if performative = "inform" [add-belief get-content msg] ;; if the performative is inform then add it to the beliefs

   ;; the inform message added in beliefs has the belief type fire, this means that the agent believes there is a fire at a certian location
   ;; and it will calculate its distance from the fire and send it back to the souter
   if exist-beliefs-of-type "fire" and current-intention = "find-target-fire" [
   let fire-beliefs beliefs-of-type "fire" ;; get all the beliefs relating to fire locations
   let closest-belief first fire-beliefs ;; get the first belief in the list
   let closest-location last closest-belief ;; initialise closest location with the location in first belief in the list
   let closest-x first closest-location ;; x coordinate of location
   let closest-y last closest-location ;; y coordinate of location

   let closest-distance (abs (xcor - closest-x)) + (abs (ycor - closest-y)) ;; calculate the distance between the fire and itself

   ;;create a reply to the original message and respond with the distance to the scouter/fire
   let reply create-reply performative msg       ;; create the message with the performative as a reply, this means receiver field will be filled in automatically
   set reply add-content closest-distance reply  ;; add the content of the message to be the distance calculated above
   send reply ;; send the reply


   ]

   ]

 update-intentions ;; update the intentions

end
```

Figure 2 Showing replying of the broadcast message with their distance

The message sent by the ground units are received by the scouters and it goes through all the messages storing the minimum distance seen and its respective message, once it has iterated over all the messages and knows the best agent to command to, it creates a reply message and the contents of the message are from a function “fire-locations [Figure 4]” and sends out the reply [Figure 3]

There are several design choices made at this point, one is that a reply is sent straight away after finding out the best agent to put out the fire and beliefs of the ground unit are used to make this possible [Figure 5 and 6] and with the help of the a newly created function [Figure 4].

The considerations were to have a similar design as ground units and messages are added to the beliefs or the best message is added to the belief and the intention would then be a function which will send out the message based on the scouters beliefs, while this would be a feasible solution, it will take away the simplicity and the argument that I had with myself is whether replying to a message should be an intention or not and the decision was in the favor of keeping it simple and straightforward. Although this would not have any implication of the performance/efficiency of the program and hence was kept this way. [Figure 3]

```
;;
;; [5] The scouter behaviour has been modified to respond to messages using the reply function from FIPA library
to collect-msg-update-intentions-sc
  ;; collect messages
;; initialisation of variables
let msg 0
let performative 0
let bestContent 10000
let content 0
let msgtoReply 0

while [not empty? incoming-queue]
  [
    ;; the responses arrived back are the distances of the ground units to the fire/scouters

   set msg get-message ;; get the message and assign it to the local variable
   set performative get-performative msg ;;get the performative and assign it to the local variable
   set content get-content msg ;; get the content of the message and assign it to the local variable


    ;; check if the content is better than all the content seen before and if it is then set it as the new best content and
    ;; record the message so it can be replied later on.
    if content < bestContent [ set bestContent content set msgtoReply msg ]

  ]

  ;; At this point the closest distance must have been selected by the loop and the scouter is ready to send out a reply
  ;; Check that indeed we have a message to reply to
  if bestContent != 10000 [

    let reply create-reply performative msgtoReply ;; create the reply message to the ground unit which is the closest
    set reply add-content fire-locations reply     ;; [6] note here that fire-locations method is called and not fire-location-s and added to the message
    send reply ;; send out the reply

    ]

  update-intentions-sc
end
```

Figure 3 Showing the altered code in collecting message and updating intentions of the scouters

The name of the list was altered in this function to distinguish between the fire coordinates requested for distance and fire coordinates assigned to the specific ground agent to extinguish the fire. [Figure 4]

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;; [7] simple data structure for fires
;;; ("fire-locations-to-put-out" (x y))
;;; returns a list that indicates the location of the fire.
to-report fire-locations
  report (list "fire-locations-to-put-out" list pxcor pycor)
end
```

Figure 4 Showing a new report mechanism added to aid the BDI design

Once the function was altered, this meant that the ground agent will have 2 types of beliefs about the world, one is about where fire is in the world and second is fire-locations-to-put-out which are the fire locations assigned to the agent, now this meant that update-intentions needed to be altered to make it work which is shown in [Figure 5 and 6].

```
;; Update intentions with respect to beliefs about the world
;; [3] Note here that belief type we are looking at is fire-locations-to-put-out and not belief of fire, the idea is simple,
;; an agent can belive there is a fire at x,y but the important factor is only that the agent has been instructed to put out the
;; fire which is done with the belief fire-locations-to-put-out
to update-intentions
  if exist-beliefs-of-type "fire-locations-to-put-out" and current-intention = "find-target-fire" ;; if there is at least one belief about a fire location
  [
     let fire-location closest-fire-location ;; get the belief that relates to the fire which is closest
     remove-belief fire-location ;; remove the belief because we will now add an intention for it

     let coords item 1 fire-location ;; get the second element of the list

     ;; In response to the fire, we need to:
     ;; 1. move towards the location of the fire
     ;; 2. once arrived, extinguish the fire
     ;; So we add intentions for each of these behaviours
     ;; Note that the intentions are pushed onto the stack in reverse order so that they are popped off of the stack in the correct order
     add-intention "put-out-fire" "fire-out" ;;2. extinguish the fire, achieved when the fire is out
     add-intention (word "move-towards-dest " coords) (word "at-dest " coords) ;; 1. move towards the location of the fire, achieved when we arrive
     add-intention "do-nothing" timeout_expired 5 ;; wait 5ms before starting (this is to help the visualisation only)
  ]
end
```

Figure 5 Showing the alterations in the update intentions of the ground units

For the update intentions to work as expected, Closest-fire-location had to be altered as well to only get the beliefs of type “fire-location-to-put-out” and once this was done, the cycle of communication was complete and the agents were able to go in different directions as they were commanded by different scouters for different fire.

```
;; Returns the belief of the closest fire location
;; [4] This is very important as it extracts the beliefs and had to be altered to only get the beliefs about fire-locations-to-put-out
;;  to only elimiate the fires commanded by the air units
to-report closest-fire-location
  let fire-beliefs beliefs-of-type "fire-locations-to-put-out" ;; get all the beliefs relating to fire locations

  let closest-belief first fire-beliefs ;; get the first belief in the list
  let closest-location last closest-belief ;; initialise closest location with the location in first belief in the list
  let closest-x first closest-location ;; x coordinate of location
  let closest-y last closest-location ;; y coordinate of location
  let closest-distance (abs (xcor - closest-x)) + (abs (ycor - closest-y)) ;; calculate Manhatten distance to current closest

  ;;iterate through the list to find the closest
```

Figure 6 Showing alterations in the closest-fire-location procedure

It was also considered on whether the scouters should start searching for next fire after reporting the fire or should stay there but the test results showed that the performance was quite similar and since I have the move-randomly for ground units, their reactiveness takes care of a lot of trees.

To wrap up the implementation process, several design choices were made throughout the design and implementation process and a lot of them don’t influence the performance or efficiency of the program and purely matter as how we perceive.

Experiments

2 Description and Justification of the experiments:

There were several tests conducted to evaluate the performance of the extended model and the experiments are as follows (Initial-water was set to 25 in all cases):

Experiment: 1:

Fire-unit-nums: 1 Tree-num: 400 Number-of-fires:40 Scouter-num:1

This experiment focused on testing what happens if there is only one unit to extinguish the fire as the worst-case scenario and to test whether it refuels when it runs out of water and general performance in terms of how randomly moving around will distinguish fires. The reason 400 trees were selected is because the agent will have a better chance of finding the trees on fire if there are a lot of them. (10% of them were on fire). Also, the communication between the scouter and ground agent can be tested the best way by only having 1 of each and tracing the communication to ensure correctness. Furthermore, this experiment will help us to evaluate the performance of the new agent vs the naïve agent we began with, the results are discussed in detail in the next section.

Experiment: 2:

Fire-unit-nums: 1 Tree-num: 100 Number-of-fires:40 Scouter-num:1

This experiment was similar to the experiment 1 but number of trees were set to 100 which was the minimum it could go and the reason for this was to see how the agent performs when the environment/trees are very sparse. This also tested how the scouter and ground unit communication work in a sparse environment and whether the decision of adding randomness helped or made it worse. These factors are very important to have a robust design as it should cope up with all kinds of environments its put it. The results of this experiment are discussed in detail in the next section.

Experiment: 3:

Fire-unit-nums: 40 Tree-num: 500 Number-of-fires:40 Scouter-num:30

This experiment was designed to test how the agents behave in a crowded environment and a maximum number of fires we can have in the simulation. The cooperation of the agents is particularly important for this experiment to work as the fires will spread very fast and the agents will not have much time if they keep blocking each others way and again, this experiment is essential to see the difference between the naïve approach and the improved agent. The results discuss this in greater detail.

Experiment: 4:

Fire-unit-nums: 40 Tree-num: 400 Number-of-fires:1 Scouter-num:30

This experiment was used to test what happens if the fires start out with 1 as its often the case with the real world that the fire starts with 1 tree and then goes wild. This was tested with 40 agents to see whether they can control it before it goes wild and becomes uncontrollable, this was kept in mind that 40 agents may not all be present at hence experiment 5 was carried out to support this. It was also a test for the scouters on whether they can they can find the fire before its too and how fast the fire spreads. This brings up the question again of whether moving randomly helped the ground units to catch the fire even before the scouts? The results are discussed in detail below.

Experiment: 5:

Fire-unit-nums: 10 Tree-num: 500 Number-of-fires:20 Scouter-num:10

This experiment was carried out to see whether if the fire starts out with only one fire and there are 10 agents deployed in the forest for precautionary measures will be able to handle it as it happens in reality that at the start very few agents(humans) try to extinguish the fire even with limited resources and they often result is saving the forest. With only 10 scouters and 10 fire units, it can be really tested on how good the distance model developed is and also in comparison with naïve agent how it performs.

Experiment: 6:

Fire-unit-nums: 10 Tree-num: 200 Number-of-fires:20 Scouter-num:5

The default setting was used for this experiment as it's a good way to compare and contrast between the naïve agent and the other agent. The parameter combination in this experiment is particularly interesting because 10% of the trees will be set on fire and there are only 5 scouters to inform the agents about the fire, without random movement of the ground agent, will they be able to perform well with the help of just 5 scouters? The results answer this themselves.

Experiment: 7:

Fire-unit-nums: 30 Tree-num: 300 Number-of-fires:20 Scouter-num:20

The parameters were chosen for this experiment as they represent almost 60% of the max values possible and would be a great benchmark between the naïve agent and the enhanced agent. It would be also good to see how the scouters behave when almost 10% of the forest is on fire and how quickly can the ground units be called to eliminate the fire. Also keeping in mind that each of the ground unit is carrying 25 units of water which makes them quite slow. The results are discussed below in detail.

Experiment: 8:

Fire-unit-nums: 40 Tree-num: 500 Number-of-fires:40 Scouter-num:25

This experiment uses almost the maximum values of the parameters whereas not really using the maximum agents, the reason for this is because this experiment is trying to see whether 40 ground units and 25 scouters can handle the maximum fire in the forest, this is crucial as 10 ground units and 10 scouters may not be available due to a technical problem such as GPS broken on the scouter or water tank leak on the ground unit, hence was a crucial requirement.

Results

3 Description and analyses of the results:

The results are an average of running the simulation **10** times to get a better estimate rather than running it a single time. (The individual results are shown at the end of the document) (Altered is the newly developed agent and naïve is what we started with)

The above experiments are ran 10 times each to get an average of the values with the help of an MS Excel sheet. The values for the unaffected trees was normalized to a percentage in some experiments to make the graph clearer as other values were quite low in such situations. Below the table provides an overview of the results obtained by running the tests 10 times each and same for the naïve agent. The table contains the average values obtained by running tests 10 times and extracting the average between the values.

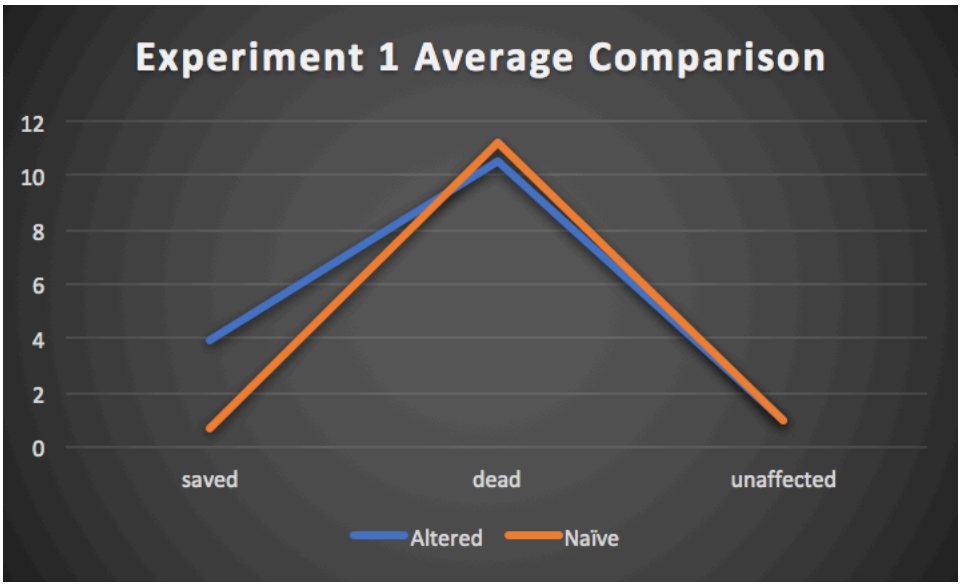
The Left value before the slash / is the improved agent and the right value is the naïve agent, the difference can be seen clearly that in all experiments improved agent has better rate of saved trees and dead trees as naïve agent.

	Saved trees	Dead Trees	Unaffected Trees
Experiment 1	3.9/0.7	10.5/11.2	385.6/388.1
Experiment 2	0.6/0.2	1.1/1.6	98.3/98.2
Experiment 3	326.7/53	0/329.1	173.3/117.9
Experiment 4	8.1/7.4	0/2.9	391.9/389.7
Experiment 5	242.2/59.2	16.9/183	240.9/257.8
Experiment 6	49.2/22.3	0.7/45.6	150.1/224.5
Experiment 7	74.1/29.2	0/45.6	225.9/224.5
Experiment 8	316.6/61.2	0/320.2	183.4/118.6

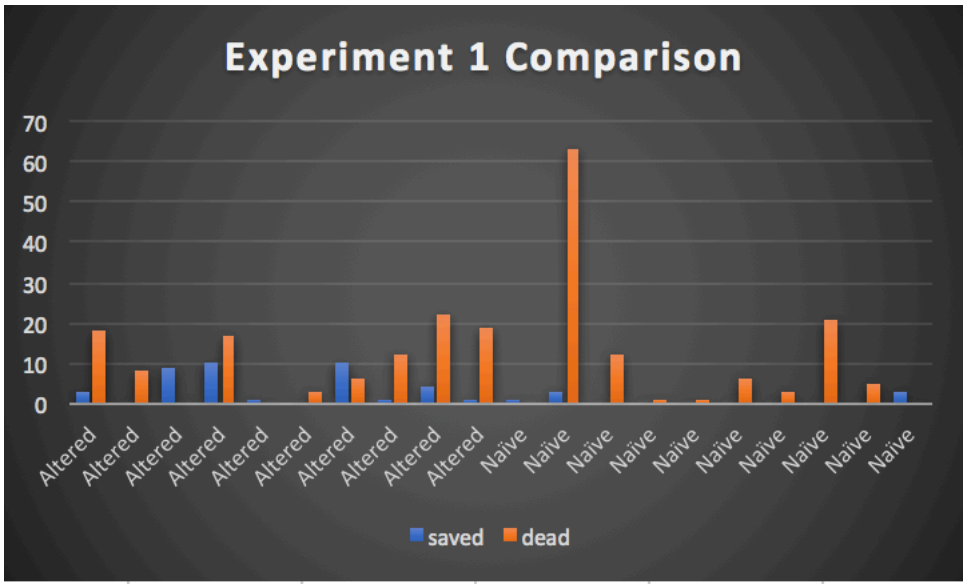
Experiment: 1:

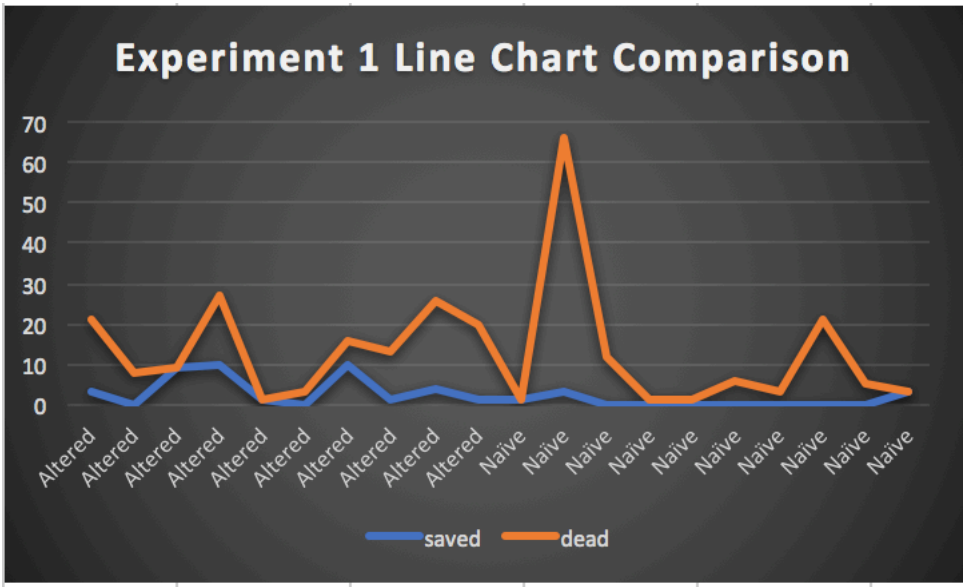
Fire-unit-nums: 1 Tree-num: 400 Number-of-fires:40 Scouter-num:1

The graphs below show that the average number of saved trees was much higher than the naïve agent, although in this case the dead trees were quite close but its very clear that the enhanced agent performs much better. One of the reasons for this is because the agent starts moving randomly at the start while the scouter is also searching for fire and this doubles the probability of finding a fire, because in either case the ground unit will try to eliminate the fire (It is possible that the tree might die while the agent is on its way to put out the fire). The results are quite reliable as they were ran 10 times for each experiment and average was taken shown in the above table and graphs below.



Below, the Bar chart and Line chart shows how each run of the experiment has yielded results and the spike can be seen in both graphs where the dead trees went quite up. Other than the spike, the results are in coordination, however the saved trees mostly higher for the altered agent as opposed to naïve approach.

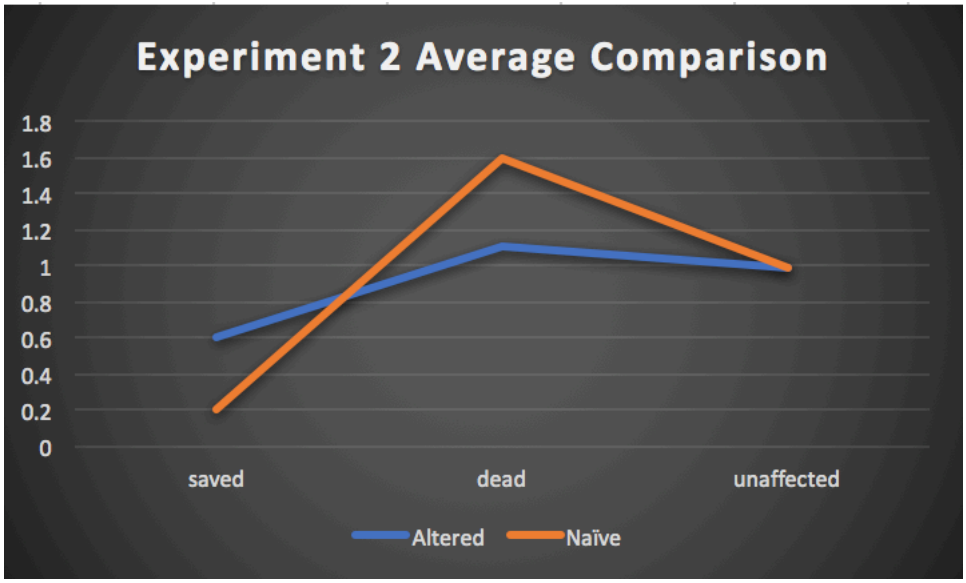




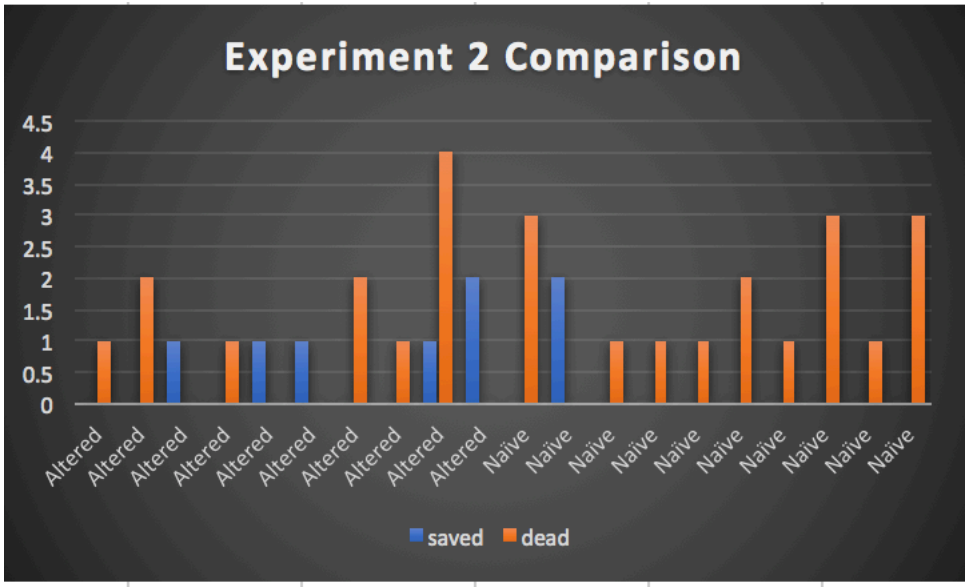
Experiment: 2:

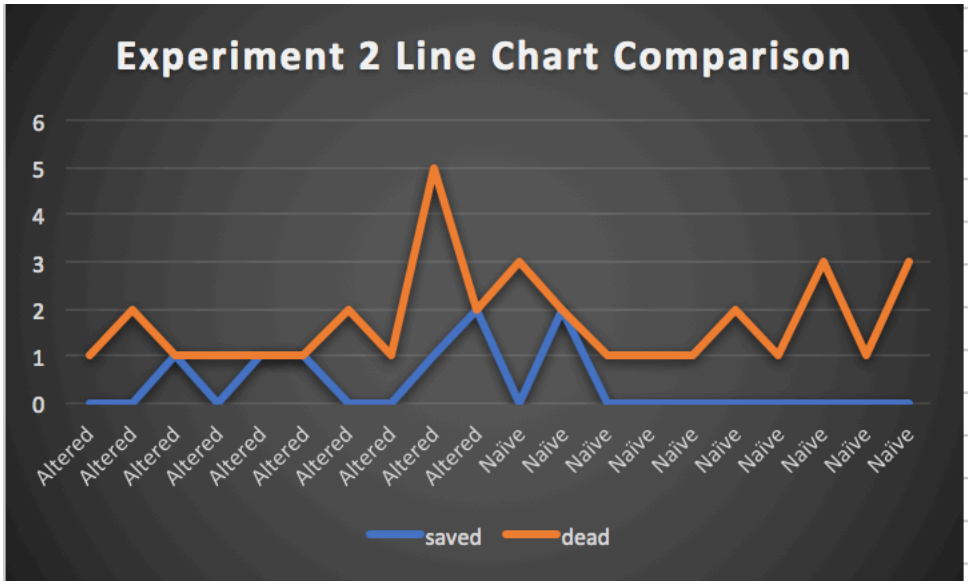
Fire-unit-nums: 1 Tree-num: 100 Number-of-fires:40 Scouter-num:1

The experiment 2 average line chart shows that the unaffected trees were more or less the same whereas the number of dead trees in the naïve approach was significantly higher than my agent and the number of saved trees were lower than the naïve agent. This means that the naïve agent performed much worse than the other agent. In this environment where almost 40% of the forest is on fire and there are only 2 agents in the whole forest (1 scouter and 1 ground unit), Both of the agents performed well as they found the fire and eliminated it before it went wild and in comparison with other experiments, this was close between the altered and naïve agent.



Below, the bar chart comparison and line chart comparison of each individual experiment shows that in some cases, the altered agent had quite a few dead trees but on the other hand it saved as many as it could and performed better than the naïve agent, and also keeping in mind the randomness of the results and the effort to make it reliable with running 10 tests each. The line chart also shows the comparison between the run of both agents.

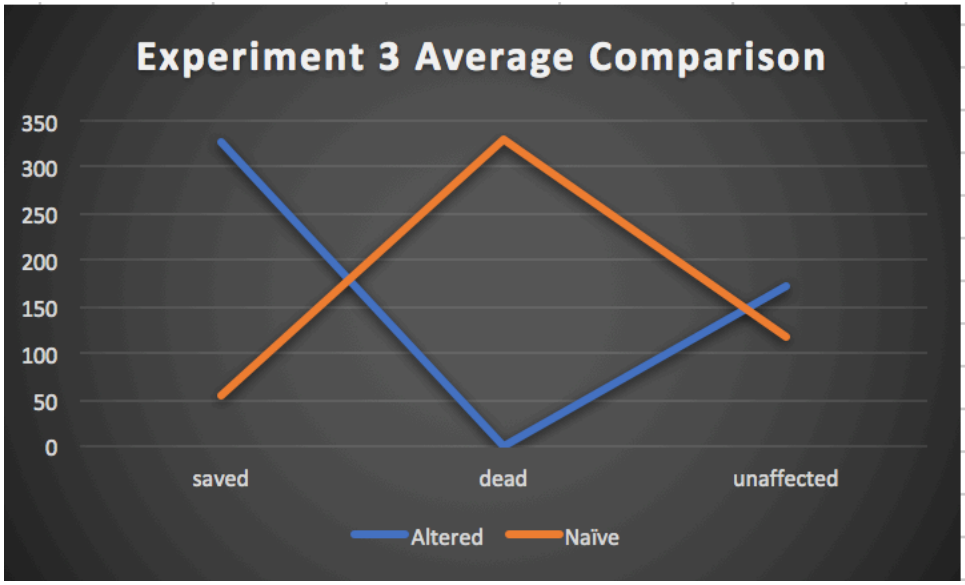




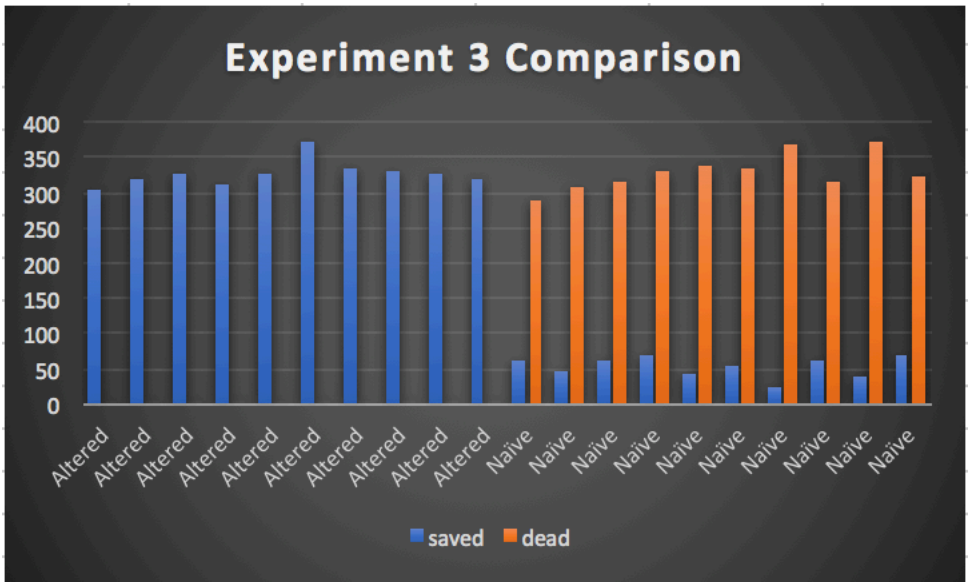
Experiment: 3:

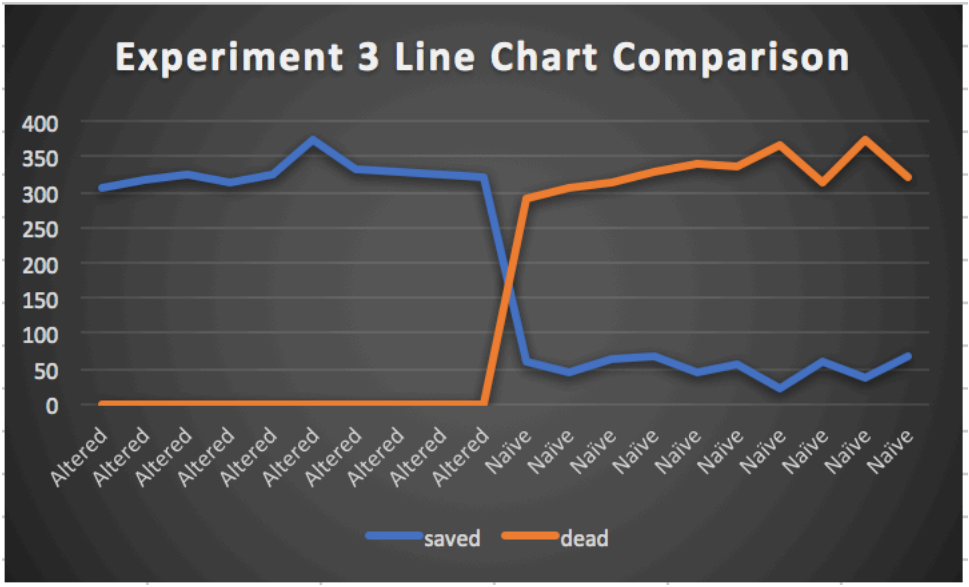
Fire-unit-nums: 40 Tree-num: 500 Number-of-fires:40 Scouter-num:30

This experiment was one the first ones to yield results which could be seen quite easily in the graph as the altered agent performed much better than the naïve agent and while the dead trees for the altered agent was 0, the naïve agent had quite a high average of 329.1 and hence the average line chart shows the angles clearly. This was particularly useful to see how random movement added played a role into saving the trees and in both cases, the number of unaffected trees were quite comparable. This shows that the altered agent was much suited to this environment as opposed to naïve agent.



Below, in the bar chart and line chart, it can be seen clearly that the altered agent had a lot more saved trees in every run as compared to the naïve agent and this is particularly clear after looking at the downhill of the blue line in the line chart and uphill of the orange line. The performance difference is also because of the naïve agent sends all agents to save 1 fire and the rest of the forest burns down whereas the altered agent only sends one agent per fire which is why the difference in performance.

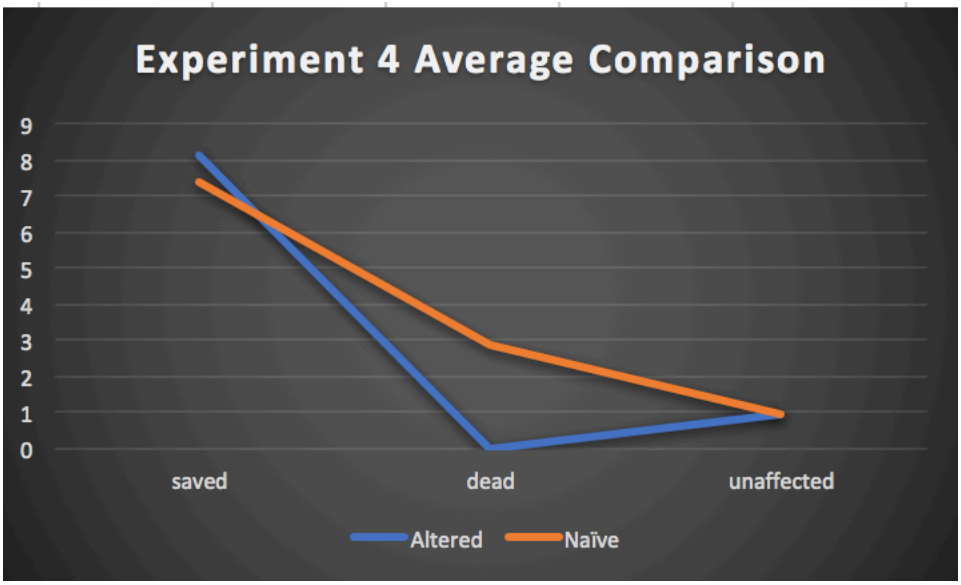




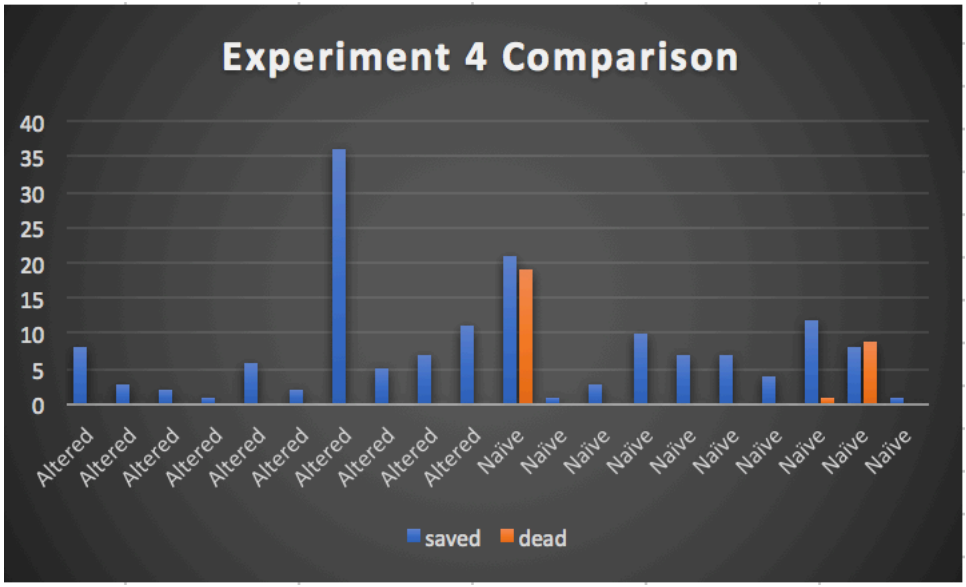
Experiment: 4:

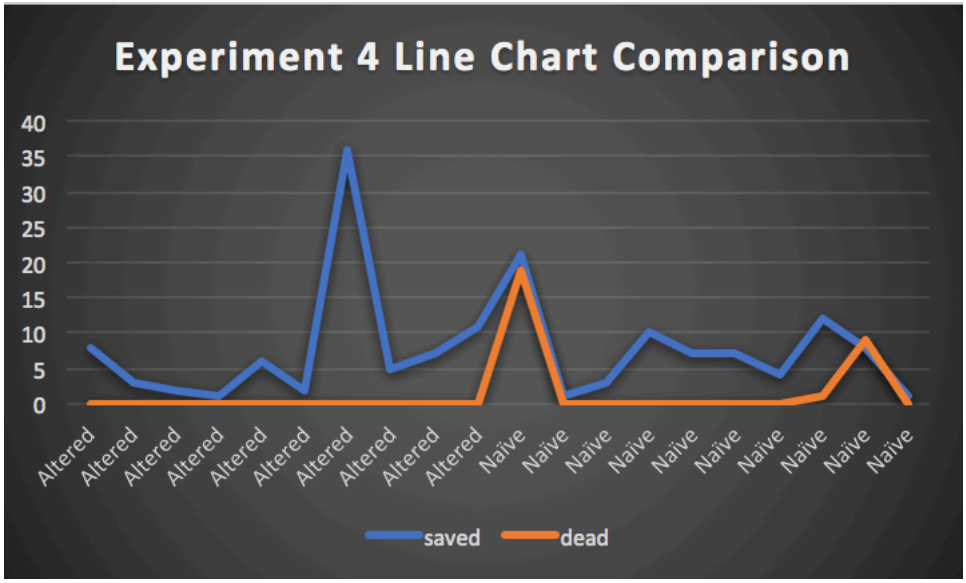
Fire-unit-nums: 40 Tree-num: 400 Number-of-fires:1 Scouter-num:30

This experiment showed that with only 1 fire starting a forest with 400 trees, the number of saved trees are about the same in both the naïve and altered agent, however the difference is with the dead trees in the altered case being 0 and being 3 in the naïve approach. This shows that with fire in only one specific place, the naïve agent can perform well, however performance of the altered was still better in this experiment as well. Individual results are shown below in bar chart and line chart.



Below, the bar chart and the line chart shows the spikes in the saved trees for altered and in the dead trees for the naïve approach. Other than that, it seems like in such an environment, the naïve agent can also perform well as it focuses on one fire and that's exactly what was needed in this experiment, However the altered agent performs even better than naïve even in this scenario.

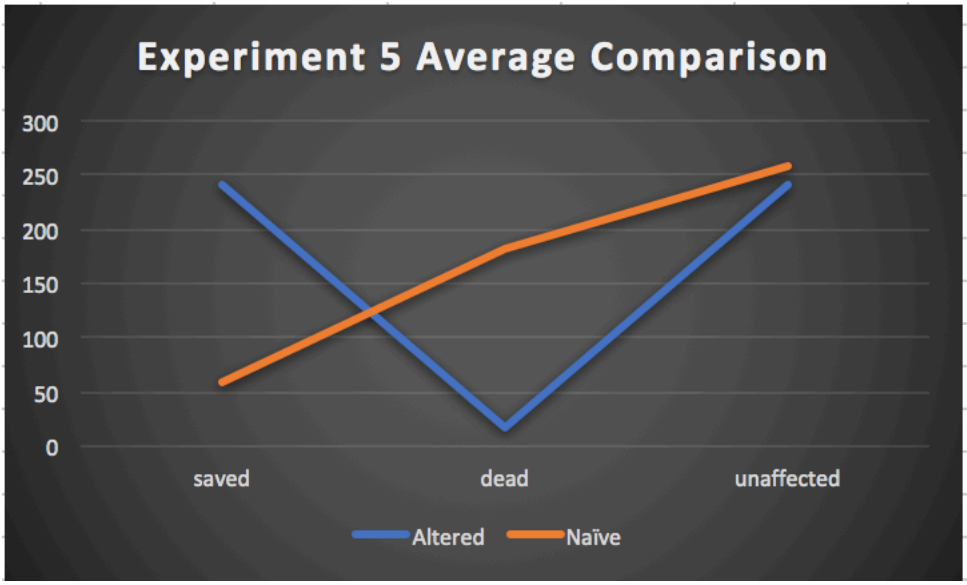




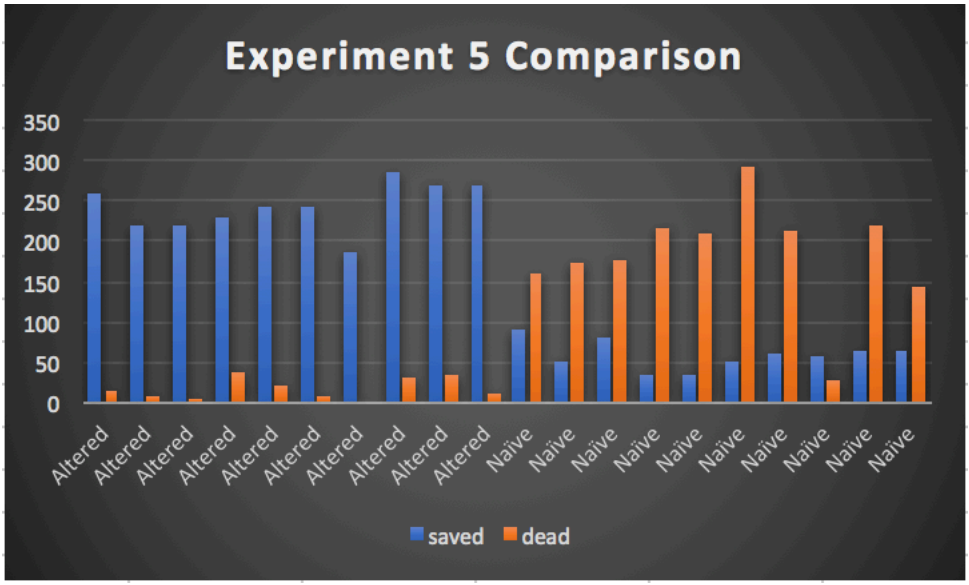
Experiment: 5:

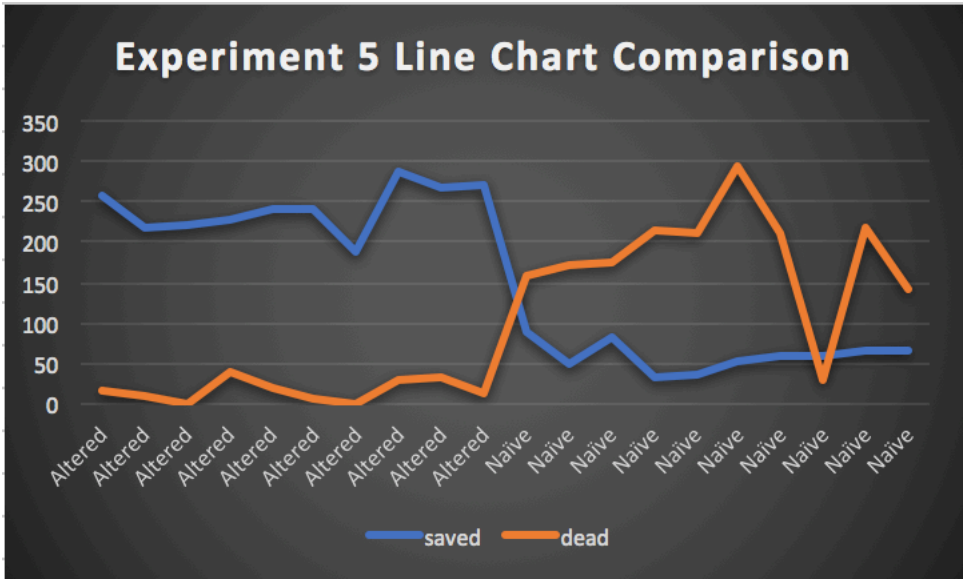
Fire-unit-nums: 10 Tree-num: 500 Number-of-fires:20 Scouter-num:10

The result of this experiment is very clear from the average graph as it can be seen that the altered makes a V having 0 dead trees and significantly more saved trees then the naïve approach. The graphs speak for themselves in this and it is very clear that the naïve agent performs really bad in an environment where the fire starts in many places as it focusses its efforts to save one tree and the rest of the forest burns down.



Below, the line chart and the bar chart shows the individual test results.

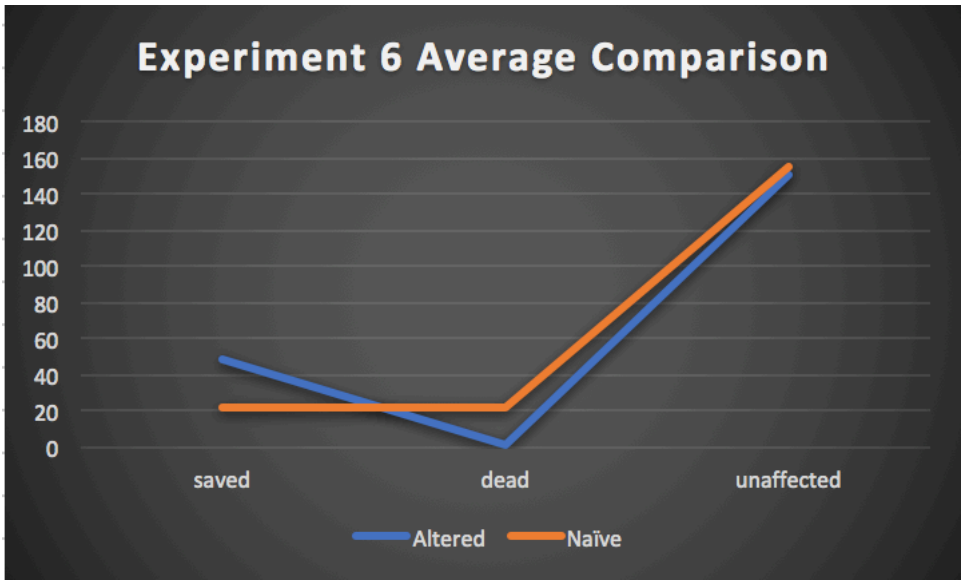




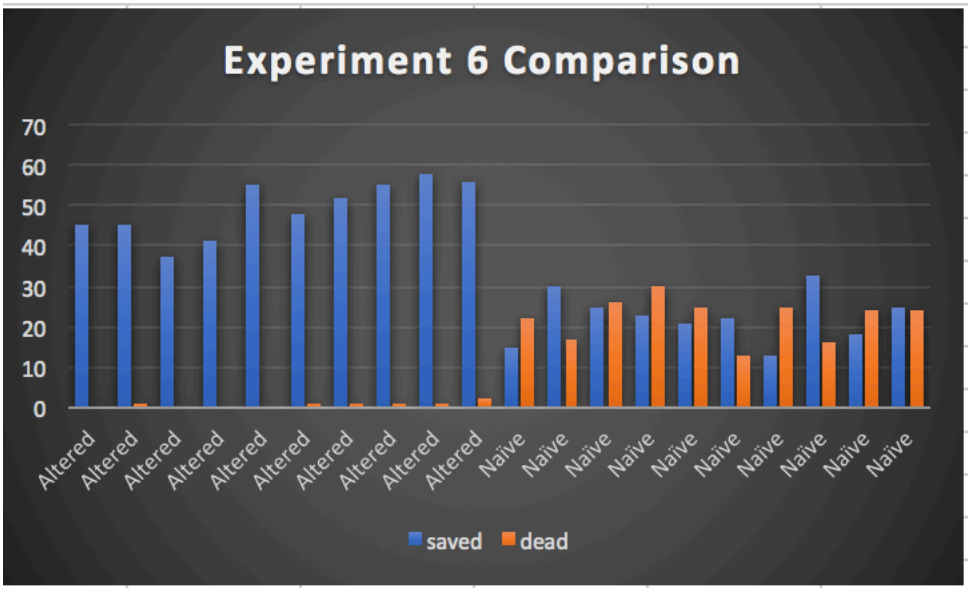
Experiment: 6:

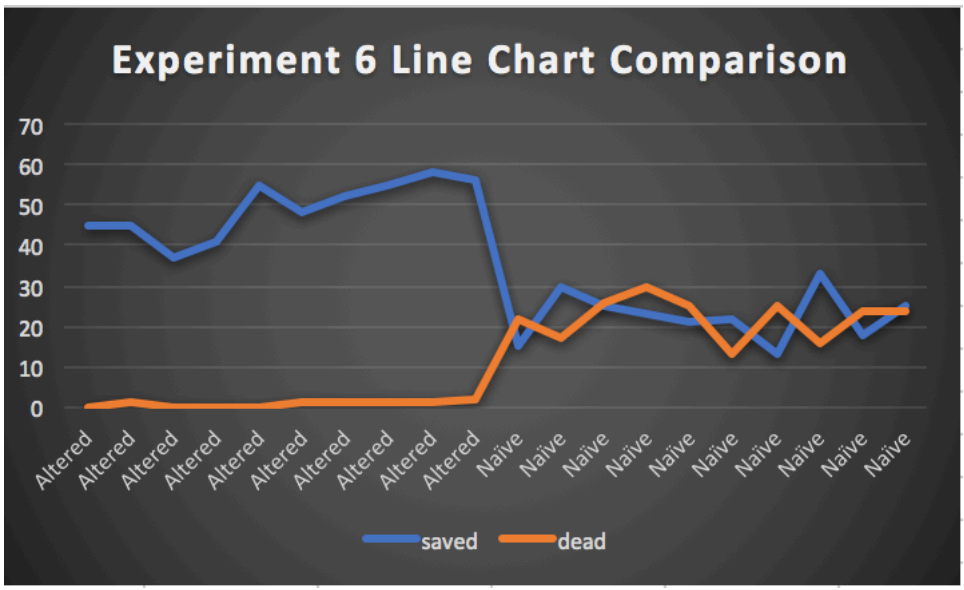
Fire-unit-nums: 10 Tree-num: 200 Number-of-fires:20 Scouter-num:5

The average comparison showed that in this experiment with only 5 scouters, the number of dead trees in the naïve agent were much higher than the altered agent and the reason for that is because altered agent will move around randomly while the scouter is also looking for fire and this increases the chances of finding any fire and also keeping in mind that in the naïve agent, it tries to eliminate all the fire in one place and the rest burns down. This can also be seen very clearly with the line chart and bar chart below.



As it can be seen in the line chart and bar chart below that the altered always had a significant number of saved trees while minimalistic number of dead trees and in comparison with naïve approach, which had almost similar dead and saved trees, the results are clearly in favor of the altered agent.

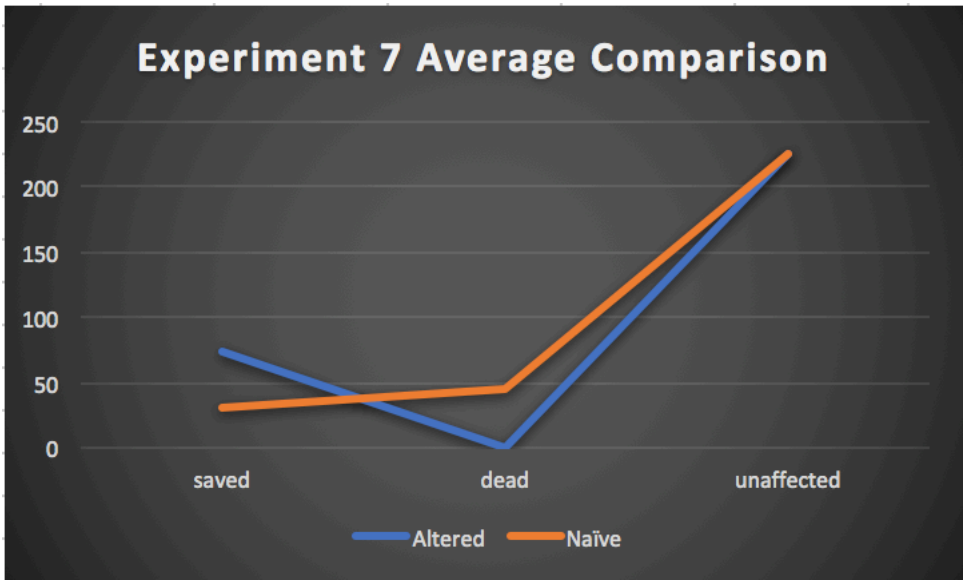




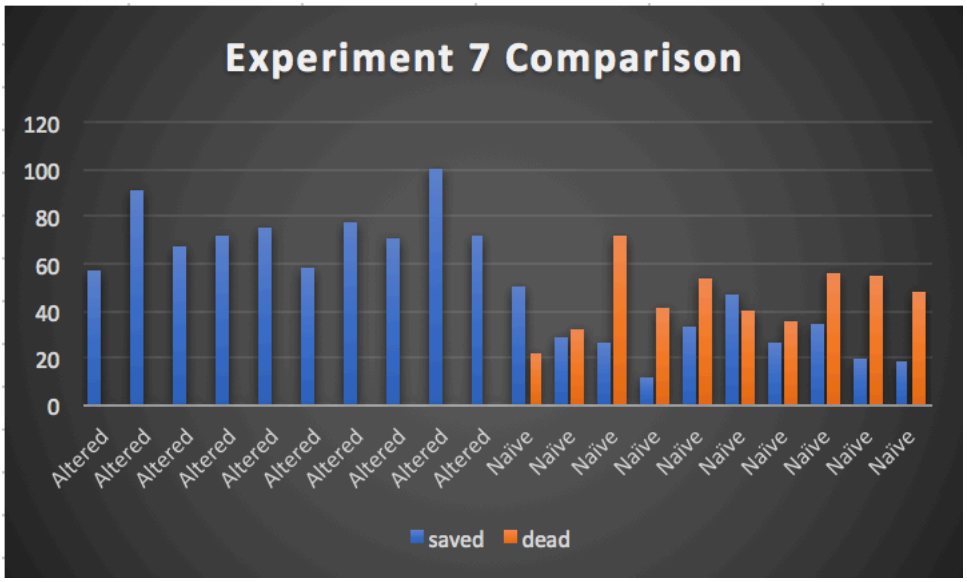
Experiment: 7:

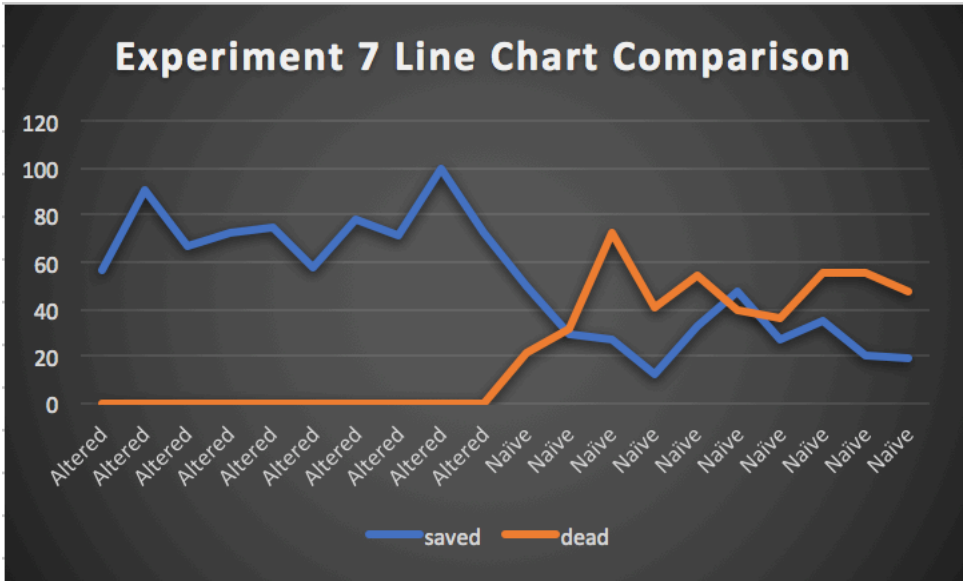
Fire-unit-nums: 30 Tree-num: 300 Number-of-fires:20 Scouter-num:20

This experiment again shows that the naïve agent had a greater number of dead trees and much less number of saved trees on average even when the unaffected was about the same. This showed that with almost half the maximum values for the parameters, the naïve agent can save quite a few trees but at the cost of several other trees dying and hence it is indeed a naïve agent.



Below, the bar chart and the line chart clearly shows that the altered agent can eliminate all the fire without letting a single tree die whereas naïve agent always has dead trees.

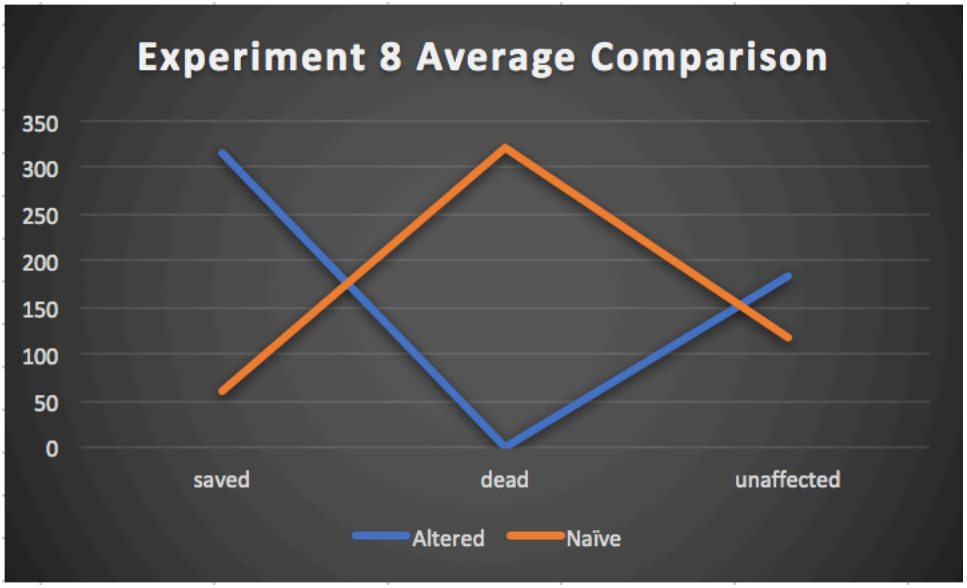




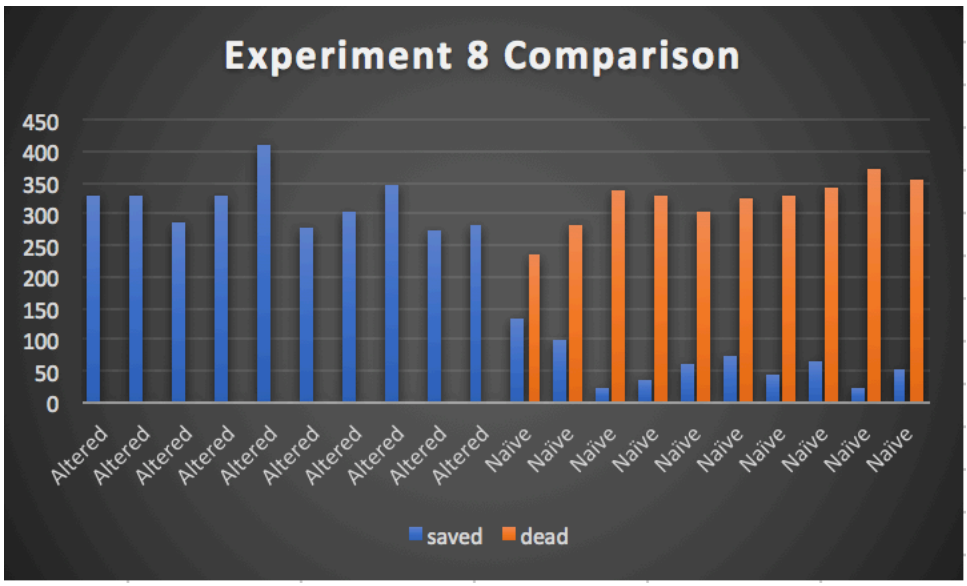
Experiment: 8:

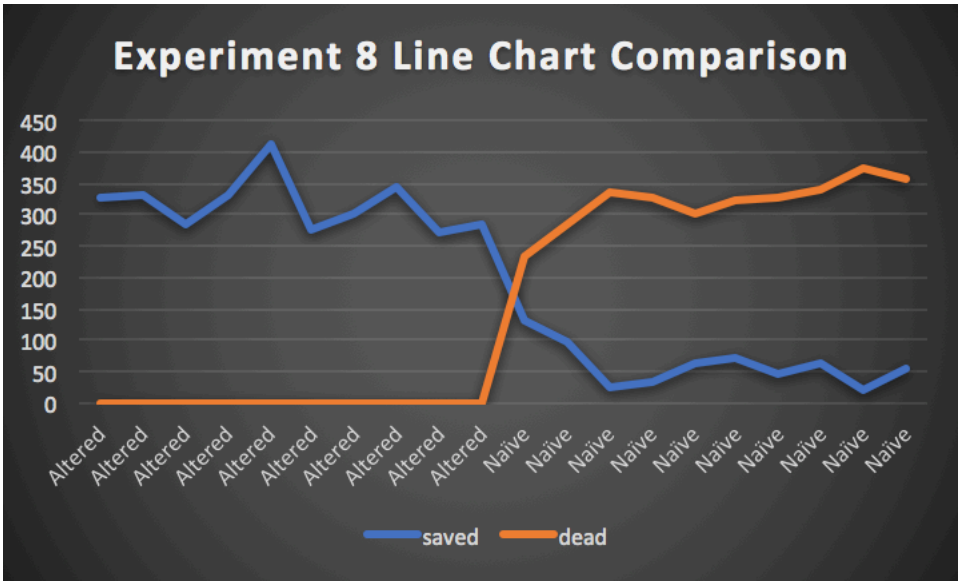
Fire-unit-nums: 40 Tree-num: 500 Number-of-fires:40 Scouter-num:25

This is one of the experiments which shows clear cut difference between the performance of the naïve agent vs altered agent as it can be seen as the floor and the sky. In this crowded environment where 40 fires will begin from random places, it was expected that the naïve agent will perform not so well but the results show that it performed much worse than expected and number of dead trees were quite high as an average while saved were much lower for naïve agent. This is a clear scenario of the improvements made and how the dead trees were brought down to 0.



Below the line chart and the bar chart shows that the dead trees for the altered agent was 0 in all experiments ran and saved trees were quite high overall, a fair comparison looking at the bar chart would be to compare altered save trees with naïve dead trees and its almost very similar whereas the saved trees for the naïve agent was much low, also, it can be said that the results are reliable to form an opinion as the experiment was ran 10 times per experiment and graphs show the stability.





Advantages

4 Advantages of Hybrid design and Problems with BDI Architecture:

There are many advantages of the hybrid approach in the forest fire problem as the forest fire starts at point and the scouters can look for the fire and inform the agent about the fire and it can be put out by the closest agent, but now considering the scenario where all agents are at the base then even though the closest agent will be asked to eliminate the fire which maybe at the further end of the forest, then without the hybrid design, the agent will ignore any fire that comes in it way and by the time it will get to the original fire reported by the scouter, the tree maybe dead and the one ignored will die too.

This short scenario explains why the hybrid design perfectly resembles the real world and it's a solution that can save forest fires. For some other domain such as parking system, reactive approach may work as the cars comes in, it issues the ticket and when they leave it takes the money based on time, otherwise it does nothing, but for the forest fire problem, a hybrid agent is necessary as the results show that it performs very well.

The first problems that will arise from using a pure BDI architecture is the one described above where the agent will follow its plan to reach a destination X and if it encounters tree T on fire on its way, it will simply ignore it and keep moving towards its target X and by the time it will reach there, X may have died and if at that point it re-plans to save T, it is likely that T will be dead by the time the agent will reach there, hence a hybrid agent is necessary for forest fire problem.

It may also be the case that the agents first plan is based on when the fire started and at that point only 1 tree is on fire whereas the next minute the whole bunch will be on fire and then the question becomes on how often will the agents re plan because if they re plan very often then that's the only thing they will do and if they do it occasionally then their knowledge about the world will be incorrect and they will not be saving as much trees as they are with the hybrid design.

I believe that the experiments have also shown that the hybrid agent works quite well for the forest fire domain and as I have mentioned above, different domains will suite different architectures and there are domains where the pure BDI architecture will work better than the hybrid agent.

Appendix/ Individual Test Results

	Experiment 1				Experiment 1 on original code		
	saved	dead	unaffected		saved	dead	unaffected
1	3	18	379	1	1	0	399
2	0	8	392	2	3	63	334
3	9	0	391	3	0	12	388
4	10	17	373	4	0	1	399
5	1	0	399	5	0	1	399
6	0	3	397	6	0	6	394
7	10	6	384	7	0	3	397
8	1	12	387	8	0	21	379
9	4	22	374	9	0	5	395
10	1	19	380	10	3	0	397
	3.9	10.5	0.964		0.7	11.2	388.1
	Experiment 2				Experiment 2 on original code		
	saved	dead	unaffected		saved	dead	unaffected
1	0	1	99	1	0	3	97
2	0	2	98	2	2	0	98
3	1	0	99	3	0	1	99
4	0	1	99	4	0	1	99
5	1	0	99	5	0	1	99
6	1	0	99	6	0	2	98
7	0	2	98	7	0	1	99
8	0	1	99	8	0	3	97
9	1	4	95	9	0	1	99
10	2	0	98	10	0	3	97
	0.6	1.1	0.983		0.2	1.6	98.2

	Experiment 3		
	saved	dead	unaffected
1	304	0	196
2	318	0	182
3	325	0	175
4	313	0	187
5	326	0	174
6	372	0	128
7	334	0	166
8	330	0	170
9	325	0	175
10	320	0	180
	326.7	0	173.3

	Experiment 3 on original code		
	saved	dead	unaffected
1	61	289	150
2	47	307	146
3	63	315	122
4	69	330	101
5	44	339	117
6	55	335	110
7	23	368	109
8	61	314	125
9	38	372	90
10	69	322	109
	53	329.1	117.9

	Experiment 5		
	saved	dead	unaffected
1	259	15	226
2	219	9	272
3	220	1	279
4	228	39	233
5	242	21	237
6	242	8	250
7	188	0	312
8	286	30	184
9	268	34	198
10	270	12	218
	242.2	16.9	240.9

	Experiment 5 on original code		
	saved	dead	unaffected
1	90	160	250
2	51	172	277
3	81	176	243
4	33	216	251
5	35	211	254
6	52	293	155
7	61	212	227
8	59	29	412
9	65	219	216
10	65	142	293
	59.2	183	257.8

	Experiment 7					Experiment 7 on original code		
	saved	dead	unaffected			saved	dead	unaffected
1	57	0	243		1	50	22	228
2	91	0	209		2	29	32	239
3	67	0	233		3	27	72	201
4	72	0	228		4	12	41	247
5	75	0	225		5	33	54	213
6	58	0	242		6	47	40	213
7	78	0	222		7	27	36	237
8	71	0	229		8	35	56	209
9	100	0	200		9	20	55	225
10	72	0	228		10	19	48	233
	74.1	0	225.9			29.9	45.6	224.5
	74.1	0	225.9			29.9	45.6	224.5
	Experiment 8					Experiment 8 on original code		
	saved	dead	unaffected			saved	dead	unaffected
1	329	0	171		1	133	234	133
2	331	0	169		2	98	283	119
3	286	0	214		3	25	336	139
4	330	0	170		4	35	327	138
5	411	0	89		5	62	303	135
6	276	0	224		6	73	324	103
7	302	0	198		7	46	327	127
8	344	0	156		8	64	340	96
9	273	0	227		9	22	373	105
10	284	0	216		10	54	355	91
	316.6	0	183.4			61.2	320.2	118.6
	316.6	0	183.4			61.2	320.2	118.6

	Experiment 4					Experiment 4 on original code		
	saved	dead	unaffected			saved	dead	unaffected
1	8	0	392		1	21	19	360
2	3	0	397		2	1	0	399
3	2	0	398		3	3	0	397
4	1	0	399		4	10	0	390
5	6	0	394		5	7	0	393
6	2	0	398		6	7	0	393
7	36	0	364		7	4	0	396
8	5	0	395		8	12	1	387
9	7	0	393		9	8	9	383
10	11	0	389		10	1	0	399
	8.1	0	0.97975			7.4	2.9	389.7

	Experiment 6					Experiment 6 on original code		
	saved	dead	unaffected			saved	dead	unaffected
1	45	0	155		1	15	22	163
2	45	1	154		2	30	17	153
3	37	0	163		3	25	26	149
4	41	0	159		4	23	30	147
5	55	0	145		5	21	25	154
6	48	1	151		6	22	13	165
7	52	1	147		7	13	25	162
8	55	1	144		8	33	16	151
9	58	1	141		9	18	24	158
10	56	2	142		10	25	24	151
	49.2	0.7	150.1			22.5	22.2	155.3
	49.2	0.7	150.1			22.5	22.2	155.3