Artificial Intelligence for Games

DAC619

sean khanna – Q11279516

Level 6

Contents

[Algorithm 2](#_Toc533773559)

[Justification 2](#_Toc533773560)

[Choice 1 2](#_Toc533773561)

[Choice 2 2](#_Toc533773562)

[Choice 3 2](#_Toc533773563)

[Comparison 2](#_Toc533773564)

[Design 2](#_Toc533773565)

[Diagrams 2](#_Toc533773566)

[Time management and organisation 2](#_Toc533773567)

[Source Control 2](#_Toc533773568)

[Other tools 2](#_Toc533773569)

[Testing 2](#_Toc533773570)

[Critical Evaluation 2](#_Toc533773571)

[Appendix 3](#_Toc533773572)

[Appendix A – HacknPlan 3](#_Toc533773573)

[Appendix B – State Diagrams 3](#_Toc533773574)

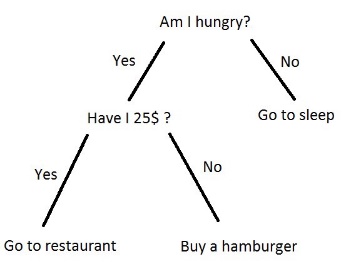
[Appendix C 3](#_Toc533773575)

# Algorithm

## Justification

Before starting this assignment, I had to choose a method of AI that I was going to implement, this meant I had to do some research into the different types of AI out there. After going through all of the labs and looking through each lecture, I was able to narrow it down to three choices that I thought were feasible for this assignment. The first was decision trees, then flocking and finally the Monte Carlo tree search.

### Decision Tree

Decision trees for me was the most basic solution for any AI; it follows a simple set of rules that is pre-maid by the developer. There is no real complexity towards it however, when showing and explaining it to anyone, it can be easily understood. If shown as a tree, as shown in picture, you can follow the behaviour of it as you go down the tree. From the top, you can see that at each intersection there is two branches that come off, yes or no, which then will lead you to another “node”; inevitably, you will reach a leaf node which could be a win, lose or draw. In this case it is going to sleep, go to restaurant or buy a hamburger. Some trees can have merging branches, this is when two or more branches lead to the same node. More intricate decision trees need a very vigilant design so that there is always a solution or leaf node.

Decision Trees, 2017. Egor Dezhic.

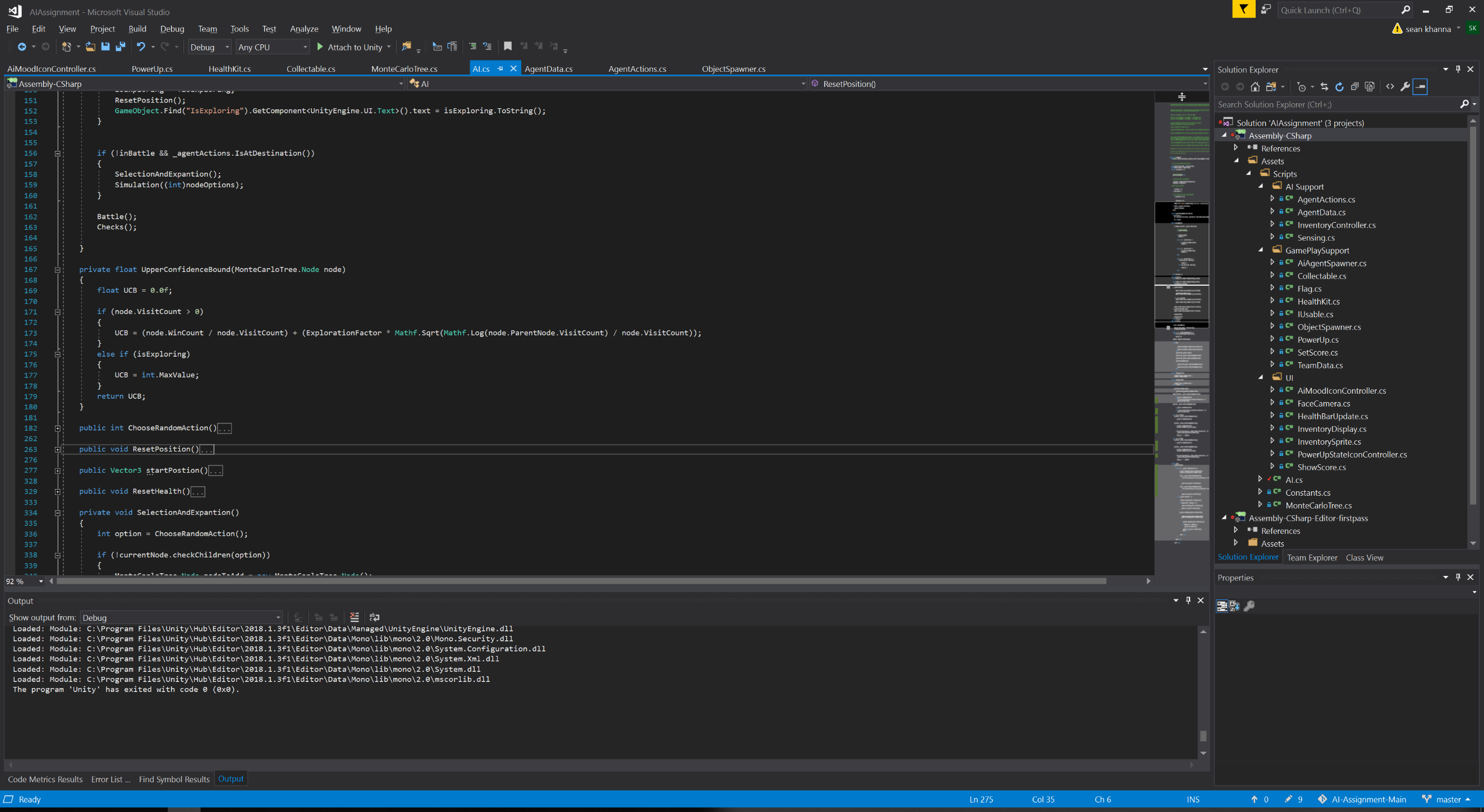
### Flocking

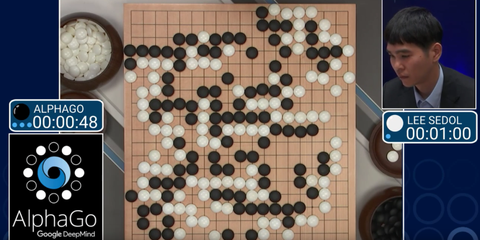
My second option was to use flocking. Flocking is very different to decision trees, flocking is more about moving in groups very similar to how animals flock like these starlings in the photo. There are three types of flocking Alignment, Cohesion and Separation. Alignment, for my game would have been the method of flocking I would have used because with the small number of team members they could have stuck together and attempt to defeat their enemies together, they would almost act as if they are teaming up. For example, if one team member spotted another team member, they would move towards them and then align with them. Unfortunately, cohesion would not have worked as well because for it to work effectively, we would need a large number of other members for it to receive better data. Finally, separation would just confuse the AI too much and they would never reach anywhere because they would always be moving away from each other, so they would never help each other it if one was attack for example.

Starlings flocking, 2011. Scott Heppell.

### Monte Carlo Tree Search

Finally, the Monte Carlo tree search, this form of AI is very complex, and it is a form of machine learning. Simply, the more time put in to this the more it learns. The Monte Carlo tree search allows each AI to make a decision either by a random factor, usually when the tree is first being created. Later, if we switch the element of exploration off, the AI will analyse the data it has and pick the best route using an algorithm called the Upper Confidence Bound.



At the beginning this tree will be empty and the value of the UCB will always be 0 for each node, so to help the algorithm build you can use a max integer value. From the root you can add one or more child nodes to the tree and allow this algorithm to pick one, if the highest UCB is the max integer value for more than one you can pick a random node to test or use a heuristic algorithm to pick one. Over some time, you will end up with either a win or lose state and then your tree will traverse back up the tree and update each node with whether or not they won and the visit count. For the assignment, if the enemy team returns the flag and wins, I would have it so that the friendly team’s tree would be updated so that it was a loss and I feel over time this would create a tree which would allow a way for the AI to get the flag quicker. The Monte Carlo tree search algorithm was used in a game called AlphaGo and is now near impossible to beat. If we took a look at the tree that has been created, it would be massive with depths of about 10000+ nodes.

AlphaGo, 2016. Jay Bennett.

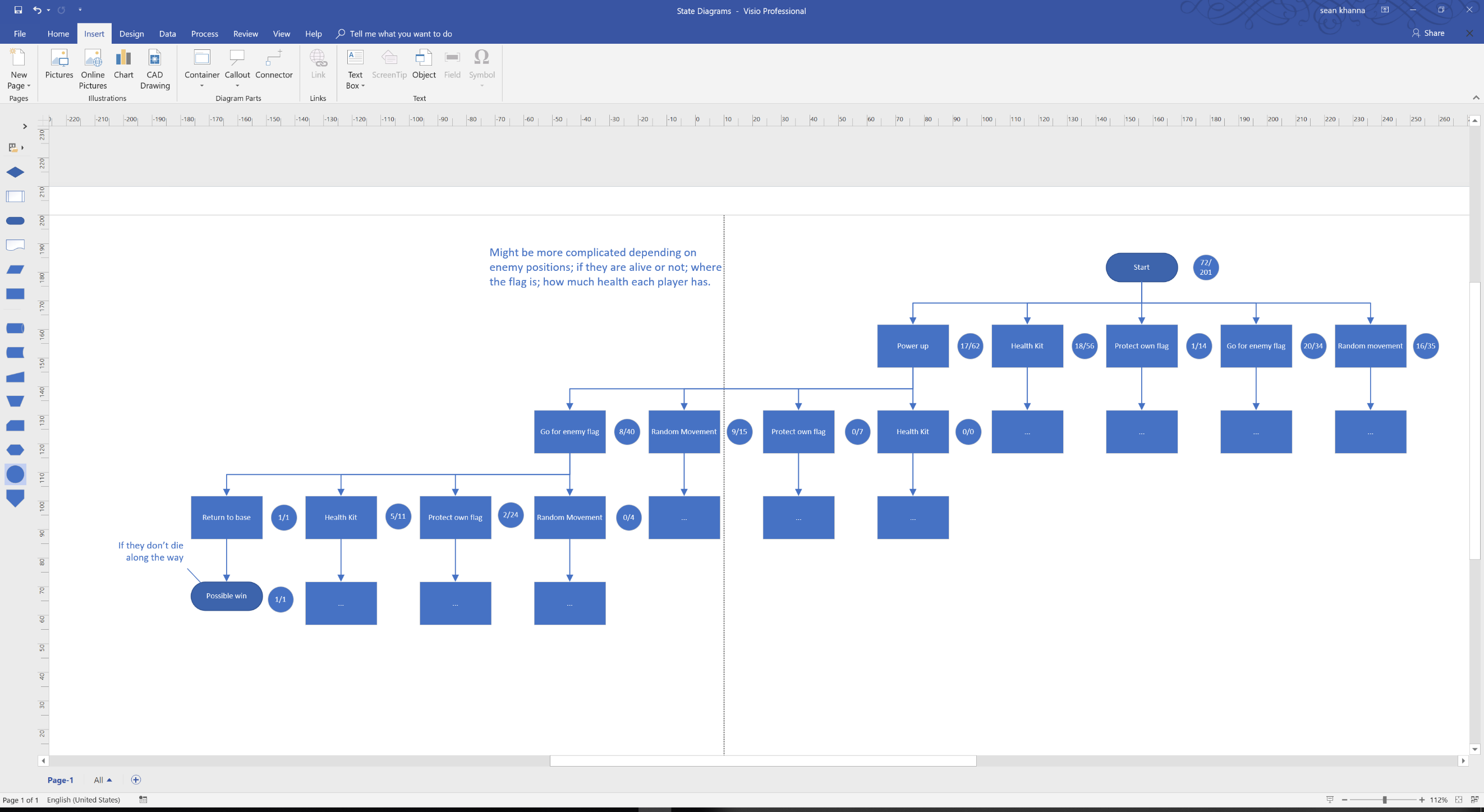
### Comparison

In the end, I decided to go with the Monte Carlo Tree Search, it appealed a lot more to me. Additionally, the Monte Carlo tree search is very good for complex games. For me, decision trees are very simple and only good for turn based games or 1 vs 1 type matches, unfortunately the game we are creating AI for, has multiple team members. Flocking on the other hand, was at the other end of the scale, in my opinion it is best suited for a game were, you would have large numbers of AI.

# Design 817

## Diagrams

### State Diagram/ Monte Carlo Tree

Each node within a Monte Carlo Tree See [Appendix B](#_Appendix_B_–)) has two values, the number of wins that have come from visiting this node and also the number of times it has been visited. Just from these two values on each child node we can calculate the UCB using the equation from before. For example, if we take two connected nodes, regardless of what they do:

As you can see the UCB is 0.26, if there is another UCB on a different node that is higher then it will choose that one instead. The higher the exploration factor, the more likely it is to explore rather than win. It will take a little experimentation to find the right value to balance exploration to exploitation.

### Flow charts 100

# Time management and organisation 100

## Source Control

## Other tools

# Testing 100

# Critical Evaluation 300

## Improvements

## Problems

# References

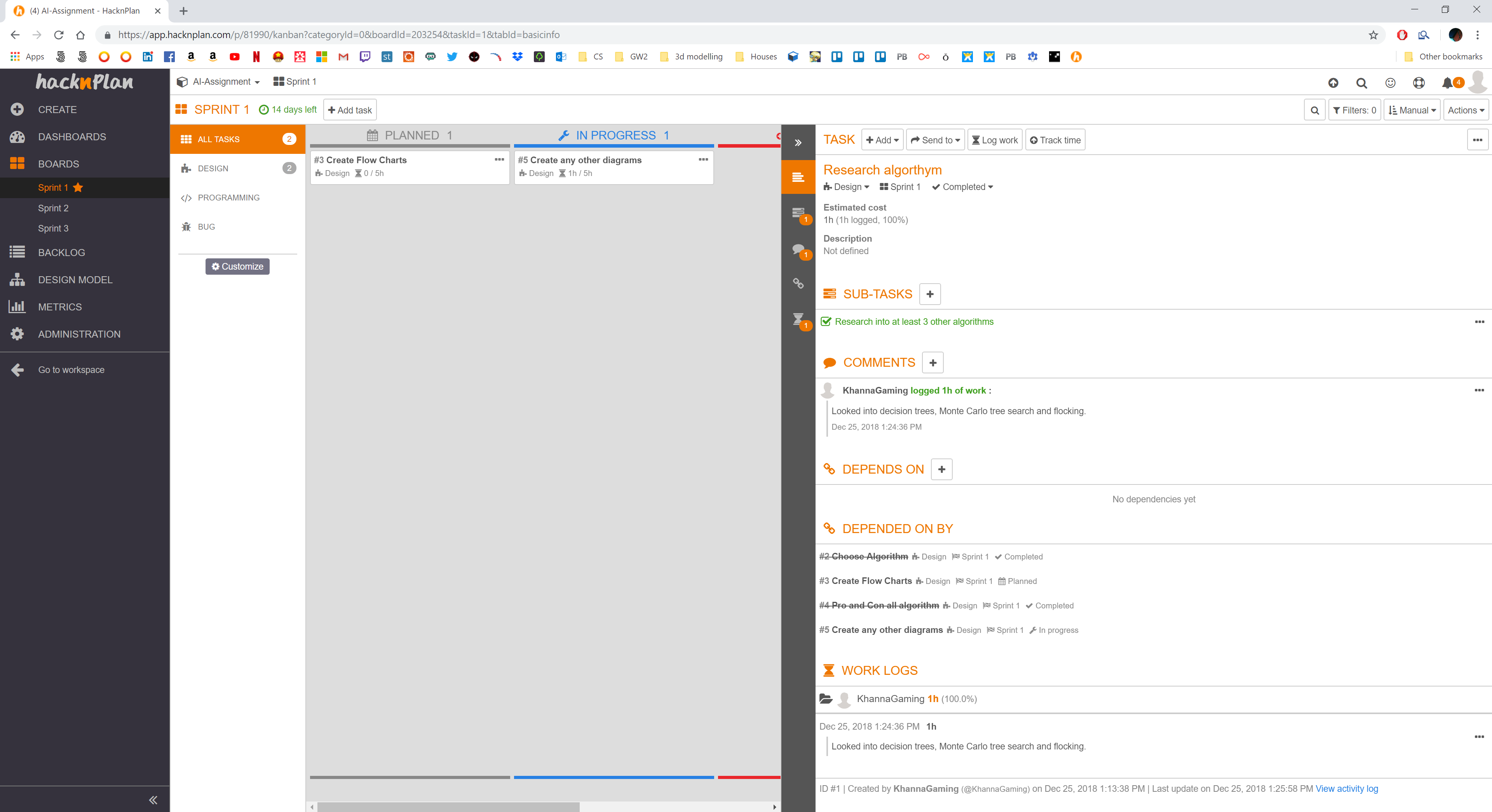
Decision Tree, 2017. Understanding Decision Trees [viewed 28/12/2018]. Available from: <https://becominghuman.ai/understanding-decision-trees-43032111380f>

Starlings flocking , 2011. Scott Heppell. A murmuration of starlings is seen in the sky as the sunset sets above Gretna, Scotland [viewed 28/12/2018]. Available from: <https://io9.gizmodo.com/you-wont-believe-the-patterns-created-by-flocks-of-bir-1469575403>

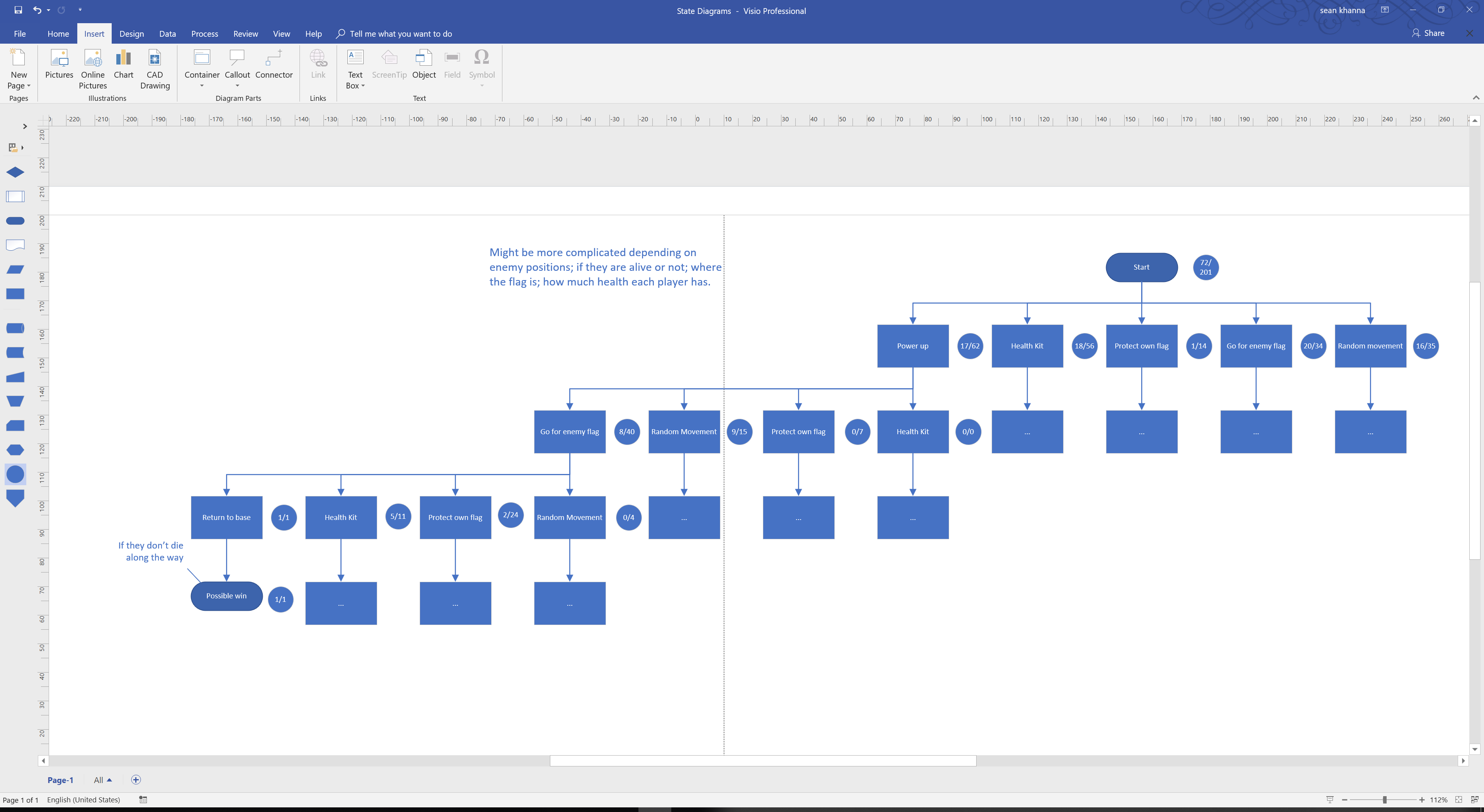
AlphaGo, 2016. Jay Bennet. Google's AlphaGo AI Continues to Wallop Expert Human Go Player [viewed 28/12/2018]. Available from: <https://www.popularmechanics.com/technology/a19863/googles-alphago-ai-wins-second-game-go/>

# Appendix

## Appendix A – HacknPlan



## Appendix B – State Diagrams



## Appendix C