SWINBURNE UNIVERSITY OF TECHNOLOGY

COS30002

Goal Oriented Behaviour

Task 4, Week 3
Spike: Spike_N004

Matthew Coulter
S102573957

26/03/2020

Tutor: Tien Pham

Table of Contents

Goals / Deliverables

The goal of this task is to learn about the implementation of Simple Goal Insistence, then improve this by implementing goal-orientated behaviour. Whilst this is introducing us to the benefits of using a slightly more advanced system, the task intends to also demonstrate the limitations of Goal Oriented Behavior. This report will contain all of the information on how to implement Goal Oriented Behavior, the benefits and limitations associated and examples used from my implementation.

Besides this report, what else was created?

For example: UML diagram, code, reports

- Code: cos30002-102573957\04 - Spike - Goal Oriented Behaviour\gob_complex\gob_complex.py
- Short report titled “ide comparison”
- GOB Diagram

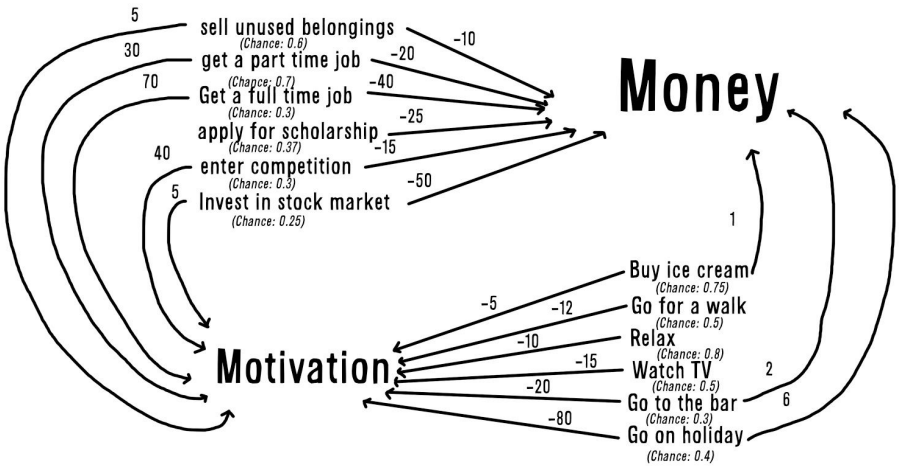
Technologies, Tools, and Resources used:

List of information needed by someone trying to reproduce this work

- Adobe Illustrator CC 2020
  For illustrating the GOB diagram
- Visual Studio 2019
  For writing the script and compiling
- Python language pack
  For error checking, debugging and compiling in Visual Studio
- Canvas Resources
  For modifying the supplied code
- Notepad++ (Not 100% necessary)
  For making quick edits such as *Replace All* or copying text from the simple version to the complex version.

Tasks undertaken:

- **Step 0:** prior to any development and planning, I would highly recommend that you know exactly what Goal oriented Behaviour is. Whilst i used the videos from Canvas, I would also recommend this video: https://www.youtube.com/watch?v=nEnNtiumgII as it is explained on a very basic but easy to understand level.
- **Step 1:** Now that you have an understanding of what GOB is, you should begin to come up with the design for the AI. It is imperative that you plan out all of the actions it will perform and the costs involved. I would also recommend having at least 2 variables that will be influenced by the actions.
- **Step 2:** Implement these onto a diagram such as the one below. I used Adobe Illustrator to design this but you can use easier more accessible software such as draw.io



Formula: (Chance x Money)/Enjoyment
**lower number is better*

- **Step 3:** Start with the basics. Extend on the provided code by adding in your custom actions and the costs involved.
- **Step 4:** Once you have made sure this is working, begin to add the negatives to each of the actions. From here, you can use a ratio between the most incessant goal and the other goal. Mine is provided in the advanced version page below.
- **Step 5:** Add the functions that calculate which option is the most efficient when reaching 0. Whilst this can be done in the action_utility function, I found it is more logical to do after calculating best_action in the choose_action function. This way, all options that work can be added to a dictionary, and the most efficient one can be calculated in a function which i called determine_cheapestfinisher.
- **Step 6:** consider any other means of AI you should implement to make the decision making process more realistic. For me this was the probability and setting maximum values on goals. Remember that the probability does not mean this option will be chosen x% of the time. It is also dependent on the current values of each established goal.

What we found out:

Simple Version:

The main characteristics of the simple version is that it doesn't consider the negatives of each action in determining which action best. In turn, it will always choose the option that produces the best possible outcome. Whilst this sounds good in theory, this has many drawbacks to the AI working functionally. The AI isn't working smartly because it isn't simulating what could practically happen. It isn't utilising all of the options that are available to reach as close to the desired outcome as possible. It will do more than what is necessary such as going on a holiday twice, instead of going once, going to the bar, going for a walk and having an ice cream.

A typical scenario may look like this:

```
>> Start <<
----------------------------------------
GOALS: {'Money': 100, 'Motivation': 100}
BEST_GOAL: Money 100
BEST ACTION: Invest in stock market
NEW GOALS: {'Money': 50, 'Motivation': 100}
----------------------------------------
GOALS: {'Money': 50, 'Motivation': 100}
BEST_GOAL: Motivation 100
BEST ACTION: Go on holiday
NEW GOALS: {'Money': 50, 'Motivation': 20}
----------------------------------------
GOALS: {'Money': 50, 'Motivation': 20}
BEST_GOAL: Money 50
BEST ACTION: Invest in stock market
NEW GOALS: {'Money': 0, 'Motivation': 20}
----------------------------------------
GOALS: {'Money': 0, 'Motivation': 20}
BEST_GOAL: Motivation 20
BEST ACTION: Go on holiday
NEW GOALS: {'Money': 0, 'Motivation': 0}
----------------------------------------
>> Done! <<
```

Advanced Version

The advanced version solves these problems through an advanced action_utility function. Originally, this function just picked the action with the most effective number to change the best_goal. However, due to the drawbacks aforementioned, the new function uses a formula to calculate a generalised number that determines the best action. This formula is as follows:

$$best\_utility = -1 * bestgoal/othergoal$$

This establishes a ratio between what is trying to be achieved and how much it costs to do so. In my case, it costs energy to make money, and it costs money to gain energy. This formula is very versatile because whether the AI is trying to prioritize the Money or Motivation, it will still do that. From here, the highest utility rating will be used to determine the best action.

Whilst this outcome is much more ideal than the simple GOB system, there are still some limitations affecting the realism of the AI. For example, the AI would always 'go for a walk' because it costs 0 money and removes 15 energy. This meant that the AI would never choose the other options because this would always become the most effective option.

To fix this, I added three extra considerations:

1. **Probability that the AI would want to do a given activity/probability the activity was successful:** For example, a probability that someone wants to go for a walk can be measured, hence, the AI wants to go for a walk 75% of the time. Furthermore, the probability of making money in the stock market can be measured. For this case, the AI has a probability of 25% to make money investing in the stock market.
2. **Putting limits on the highest possible values for each goal:** By capping both money and energy at 200, no option will be chosen if the result has an effect over the other goal of over 200.
3. **Choosing the appropriate solution:** In some cases, there may be multiple solutions that will drive the goal to 0 or below. Hence, the appropriate option that lands the AI as close to under 0 as possible is chosen. This seems more realistic as a human would not go overboard if it is not necessary.

A given scenario can now look like this: [NEXT PAGE]

```
>> Start <<
-----------------------------------
GOALS: {'Money': 100, 'Motivation': 100}
BEST_GOAL: Money 100
BEST ACTION: Apply for scholarship
NEW GOALS: {'Money': 75, 'Motivation': 100}
-----------------------------------
GOALS: {'Money': 75, 'Motivation': 100}
BEST_GOAL: Motivation 100
BEST ACTION: Watch TV
NEW GOALS: {'Money': 75, 'Motivation': 85}
-----------------------------------
GOALS: {'Money': 75, 'Motivation': 85}
BEST_GOAL: Motivation 85
BEST ACTION: Go for a walk
NEW GOALS: {'Money': 75, 'Motivation': 73}
-----------------------------------
GOALS: {'Money': 75, 'Motivation': 73}
BEST_GOAL: Money 75
BEST ACTION: Get a part time job
NEW GOALS: {'Money': 55, 'Motivation': 103}
-----------------------------------
GOALS: {'Money': 55, 'Motivation': 103}
BEST_GOAL: Motivation 103
BEST ACTION: Watch TV
NEW GOALS: {'Money': 55, 'Motivation': 88}
-----------------------------------
GOALS: {'Money': 55, 'Motivation': 88}
BEST_GOAL: Motivation 88
BEST ACTION: Go for a walk
NEW GOALS: {'Money': 55, 'Motivation': 76}
-----------------------------------
GOALS: {'Money': 55, 'Motivation': 76}
BEST_GOAL: Motivation 76
BEST ACTION: Go on holiday
NEW GOALS: {'Money': 61, 'Motivation': 0}
-----------------------------------
GOALS: {'Money': 61, 'Motivation': 0}
BEST_GOAL: Money 61
BEST ACTION: Sell unused belongings
NEW GOALS: {'Money': 51, 'Motivation': 5}
-----------------------------------
GOALS: {'Money': 51, 'Motivation': 5}
BEST_GOAL: Money 51
BEST ACTION: Get a part time job
NEW GOALS: {'Money': 31, 'Motivation': 35}
-----------------------------------
GOALS: {'Money': 31, 'Motivation': 35}
BEST_GOAL: Motivation 35
BEST ACTION: Watch TV
NEW GOALS: {'Money': 31, 'Motivation': 20}
-----------------------------------
GOALS: {'Money': 31, 'Motivation': 20}
BEST_GOAL: Money 31
BEST ACTION: Apply for scholarship
NEW GOALS: {'Money': 6, 'Motivation': 20}
-----------------------------------
GOALS: {'Money': 6, 'Motivation': 20}
BEST_GOAL: Motivation 20
BEST ACTION: Go for a walk
NEW GOALS: {'Money': 6, 'Motivation': 8}
-----------------------------------
GOALS: {'Money': 6, 'Motivation': 8}
BEST_GOAL: Motivation 8
BEST ACTION: Go for a walk
NEW GOALS: {'Money': 6, 'Motivation': 0}
-----------------------------------
GOALS: {'Money': 6, 'Motivation': 0}
BEST_GOAL: Money 6
BEST ACTION: Sell unused belongings
NEW GOALS: {'Money': 0, 'Motivation': 5}
-----------------------------------
GOALS: {'Money': 0, 'Motivation': 5}
BEST_GOAL: Motivation 5
BEST ACTION: Go for a walk
NEW GOALS: {'Money': 0, 'Motivation': 0}
-----------------------------------
>> Done! <<
```

It should be noticed that the contour and variety are much more realistic; switching from money focussed to motivation focussed. Whilst the simplified version did this, this AI chooses a variety of options and because of the negative effects, the AI has a lot more decision-making to process in order to make the right decision which in turn makes the day last longer.

Open issues/risks

The software requires to enter a value of 0 for money or 0 for motivation, despite leaving it blank should imply this. This can be fixed but I don't believe it is of importance for the purpose of this task.

Recommendations

This isn't as much of a problem as it is a feature: because of the probability implementation, the AI won't always choose the BEST outcome. However, this is my intention. If you look at the complex version ending:

```
GOALS: {'Money': 0, 'Energy': 5}
BEST_GOAL: Energy 5
BEST ACTION: Go for a walk
NEW GOALS: {'Money': 0, 'Energy': 0}
```

The AI chose to 'Go for a walk' even though 'Relax' would have achieved the same outcome but be closer to 0:

```
'Go for a walk': { 'Money': 0, 'Energy': -12 },
'Relax': { 'Money': 0, 'Energy': -10 },
```

However, the reason I believe this is not a problem is because the probability was implemented for the prime purpose of simulating human wants. The AI technically considered relaxing, but preferred to go for a walk.