Spike: 7
Title: Goal Oriented Action Planning

Author: Ryan Chessum - 102564760

Goals / deliverables:

Create a GOAP simulation that demonstrates the effectiveness of the technique in considering long-term outcomes of actions (related to side-effects and/or time delays) and can plan and act intelligently.

Technologies, Tools, and Resources used:

- Visual Studio Code
- Python 3.8.0
- Lectures
- Lecture Notes

Tasks undertaken:

- Research GOAP
- Create a GOAP simulation
- Try the example using different
- Measure the big O notation of the example
- Compare the effectiveness of the simulation at different planning depths

What we found out:

GOAP is a model of GOB that creates multiple plan paths for every action it can take then compares each path to see which is the most optimal path to take.

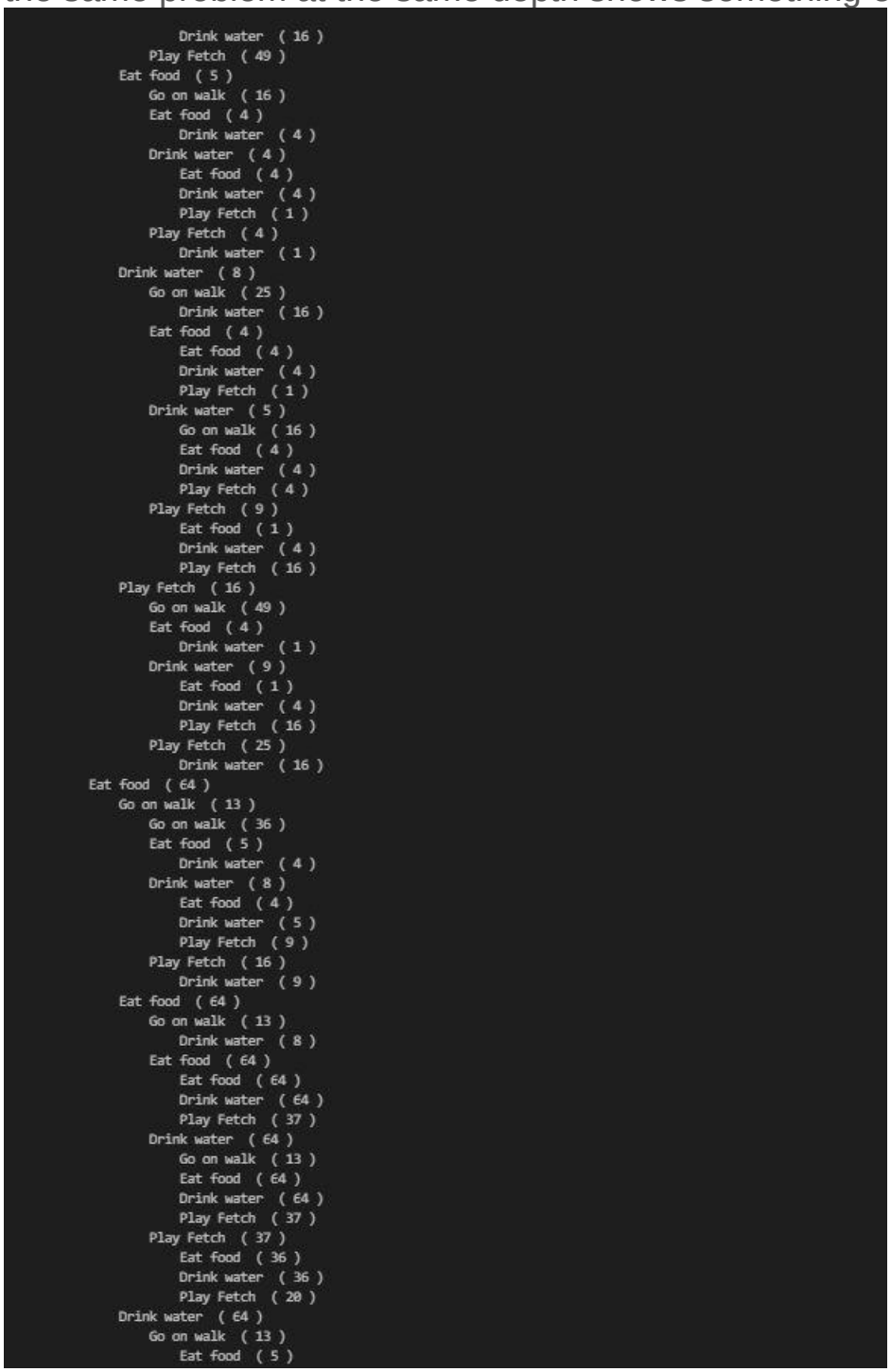
In my simulation I made a GOAP instance of the day in the life of a dog.

```
def dog_goap():
    #list that is then converted to dictionary
    goals = {
        Goal('Food', 1, 0),
        Goal('Enjoyment', 8, 0),
    }

    goals = {g.name: g for g in goals}

    actions = {
        Action('Play Fetch', goals={'Enjoyment': -2, 'Food': 1}, costs={'Time in day': -2}),
        Action('Go on walk', goals={'Enjoyment': -6, 'Food': 3}, costs={'Time in day': -3}),
        Action('Drink water', goals={'Food': -1}, costs={'Time in day': -1}),
        Action('Eat food', goals={'Food': -2}, costs={'Time in day': -2}),
    }
```

At a depth of 2 the model is able to find the optimal day in 5 steps

```
  Base  ( 65 )
      Drink water  ( 64 )
      Eat food  ( 64 )
      Go on walk  ( 20 )
      Play Fetch  ( 40 )
Best Action:
 = Go on walk (20)
New Goals (Discontentment=20)
Food=4 - , Enjoyment=2
------------------------------------------------------------
Current Goals (Discontentment= 20)
 -  Food=4, Enjoyment=2
Current Costs {'Time in day': 7, 'time': 0}
Searching...
  Base  ( 20 )
      Drink water  ( 13 )
      Eat food  ( 8 )
      Go on walk  ( 49 )
      Play Fetch  ( 25 )
Best Action:
 = Eat food (8)
New Goals (Discontentment=8)
Food=2 - , Enjoyment=2
------------------------------------------------------------
Current Goals (Discontentment= 8)
 -  Food=2, Enjoyment=2
Current Costs {'Time in day': 5, 'time': 0}
Searching...
  Base  ( 8 )
      Drink water  ( 5 )
      Eat food  ( 4 )
      Go on walk  ( 25 )
      Play Fetch  ( 9 )
Best Action:
 = Eat food (4)
New Goals (Discontentment=4)
Food=0 - , Enjoyment=2
------------------------------------------------------------
Current Goals (Discontentment= 4)
 -  Food=0, Enjoyment=2
Current Costs {'Time in day': 3, 'time': 0}
Searching...
  Base  ( 4 )
      Drink water  ( 4 )
      Eat food  ( 4 )
      Go on walk  ( 9 )
      Play Fetch  ( 1 )
Best Action:
 = Play Fetch (1)
New Goals (Discontentment=1)
Food=1 - , Enjoyment=0
------------------------------------------------------------
Current Goals (Discontentment= 1)
 -  Food=1, Enjoyment=0
Current Costs {'Time in day': 1, 'time': 0}
Searching...
  Base  ( 1 )
      Drink water  ( 0 )
Best Action:
 = Drink water (0)
New Goals (Discontentment=0)
Food=0 - , Enjoyment=0
------------------------------------------------------------
Day over!
------------------------------------------------------------
>> Done! <<
```

This is a nice method for finding the right set of actions to take but running the same problem at the same depth shows something else.



The program takes a massive amount of time to find the best action and it still uses the same 5 steps.

This model has a big O notation of O(n^d) where n is the number of actions and d is the depth of the search.

Though it is great when the depth is small and it is applied for creating shorter term plans for problems, the cost of a GOAP instance can quickly become expensive the more you want to plan into the future.

It could potentially have real world applications but with video game ai you will often be calling a behaviour many times a second. With this much complexity it doesn’t have an immediate application.

Perhaps an ai could call it once in an interval of time to come up with a short term plan, then follow that plan until the next interval of time.