SPIKE REPORT LAB 03

Jake Lucic

2103435 AI for Games

Number: Lab 03
Spike Title: Goal Oriented Behaviour with Simple Goal Insistence
Personal: Jake Lucic (2103435)

Goals:
Create a simple goal insistence (SGI) model simulation of goal-oriented behaviour (GOB) that demonstrates the both the effectiveness and the limitations of the technique. You will need to deliver (show your tutor) the following items:

- Working code that simulates and displays GOB using SGI. (There is some code provided but you don't need to use it if you don't want to.)
  - You must demonstrate a situation where SGI works appropriately
  - You must demonstrate a situation where SGI does not work well.
- Spike Outcome Report. (Use template headings. Your report should include snippets of code and results of demonstrations.)

Technologies, Tools, and Resources used:

In this task, the technologies used are listed below:

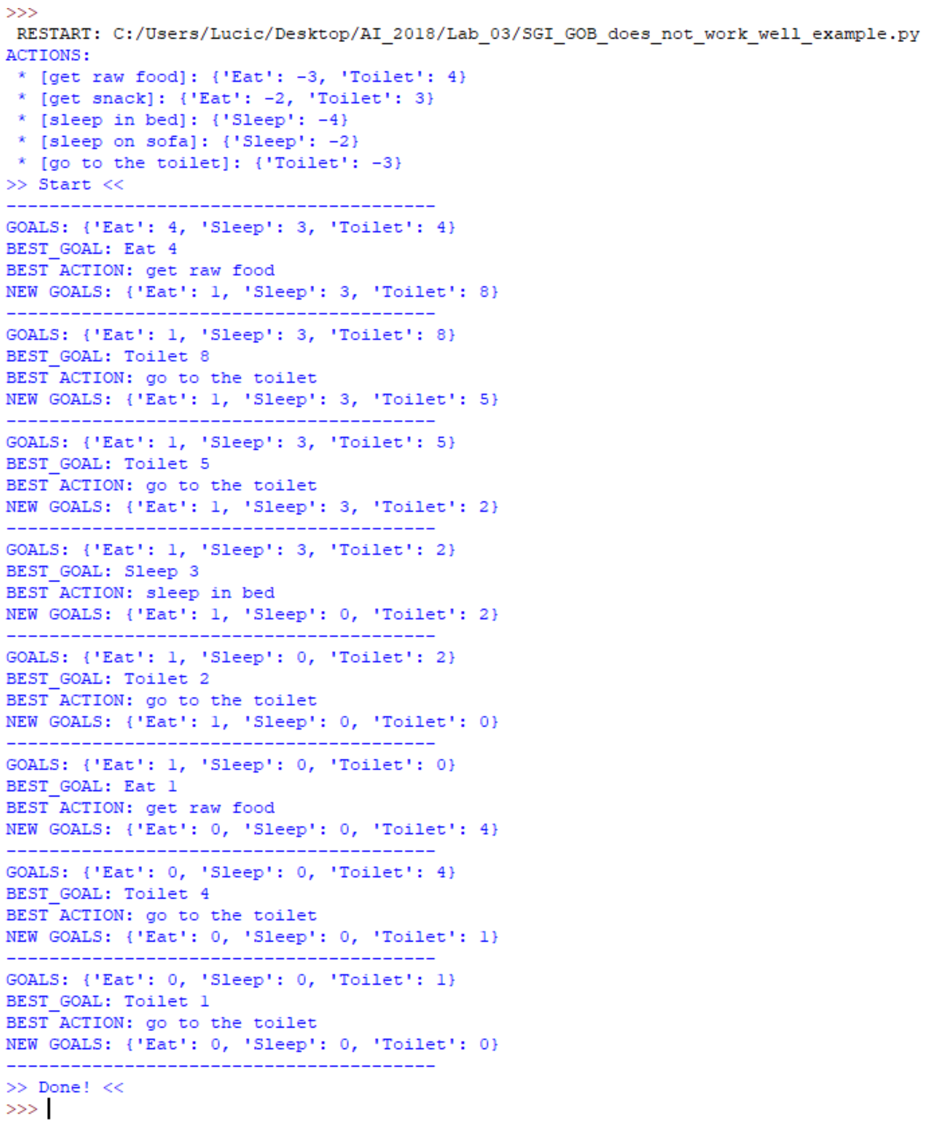
- Simple Code base, the not-yet-functional code that required fixing.
- Python IDLE v3.6.4 / Python language
- "*Artificial Intelligence for Games*" Book by Ian Millington and John Fung. *Chapter 5*

Tasks undertaken:

- Read the code base and understand what each function does
- Study the different key words and variable names
- Read chapter 5 of "*Artificial Intelligence for Games*" Book and compare to task.
- Fill in the blanks that are necessary to make code functional
- Create two working versions that display AI behaviour using SGI

What I found out:

Code A = Simple SGI using the most pressing goal (not very intelligent behaviour)
Code B = Discontentment driven SGI (more intelligent behaviour)

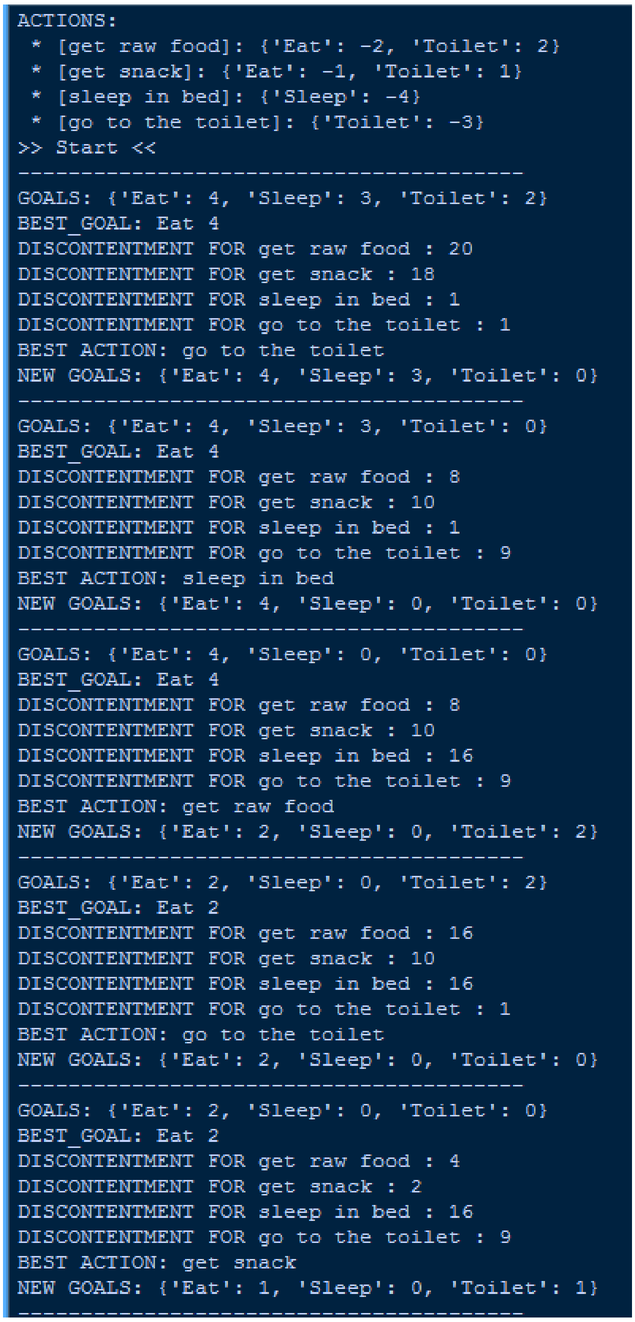


```
>>>
 RESTART: C:/Users/Lucic/Desktop/AI_2018/Lab_03/SGI_GOB_does_not_work_well_example.py
ACTIONS:
 * [get raw food]: {'Eat': -3, 'Toilet': 4}
 * [get snack]: {'Eat': -2, 'Toilet': 3}
 * [sleep in bed]: {'Sleep': -4}
 * [sleep on sofa]: {'Sleep': -2}
 * [go to the toilet]: {'Toilet': -3}
>> Start <<
----------------------------------------
GOALS: {'Eat': 4, 'Sleep': 3, 'Toilet': 4}
BEST_GOAL: Eat 4
BEST_ACTION: get raw food
NEW GOALS: {'Eat': 1, 'Sleep': 3, 'Toilet': 8}
----------------------------------------
GOALS: {'Eat': 1, 'Sleep': 3, 'Toilet': 8}
BEST_GOAL: Toilet 8
BEST_ACTION: go to the toilet
NEW GOALS: {'Eat': 1, 'Sleep': 3, 'Toilet': 5}
----------------------------------------
GOALS: {'Eat': 1, 'Sleep': 3, 'Toilet': 5}
BEST_GOAL: Toilet 5
BEST_ACTION: go to the toilet
NEW GOALS: {'Eat': 1, 'Sleep': 3, 'Toilet': 2}
----------------------------------------
GOALS: {'Eat': 1, 'Sleep': 3, 'Toilet': 2}
BEST_GOAL: Sleep 3
BEST_ACTION: sleep in bed
NEW GOALS: {'Eat': 1, 'Sleep': 0, 'Toilet': 2}
----------------------------------------
GOALS: {'Eat': 1, 'Sleep': 0, 'Toilet': 2}
BEST_GOAL: Toilet 2
BEST_ACTION: go to the toilet
NEW GOALS: {'Eat': 1, 'Sleep': 0, 'Toilet': 0}
----------------------------------------
GOALS: {'Eat': 1, 'Sleep': 0, 'Toilet': 0}
BEST_GOAL: Eat 1
BEST_ACTION: get raw food
NEW GOALS: {'Eat': 0, 'Sleep': 0, 'Toilet': 4}
----------------------------------------
GOALS: {'Eat': 0, 'Sleep': 0, 'Toilet': 4}
BEST_GOAL: Toilet 4
BEST_ACTION: go to the toilet
NEW GOALS: {'Eat': 0, 'Sleep': 0, 'Toilet': 1}
----------------------------------------
GOALS: {'Eat': 0, 'Sleep': 0, 'Toilet': 1}
BEST_GOAL: Toilet 1
BEST_ACTION: go to the toilet
NEW GOALS: {'Eat': 0, 'Sleep': 0, 'Toilet': 0}
----------------------------------------
>> Done! <<
>>> |
```

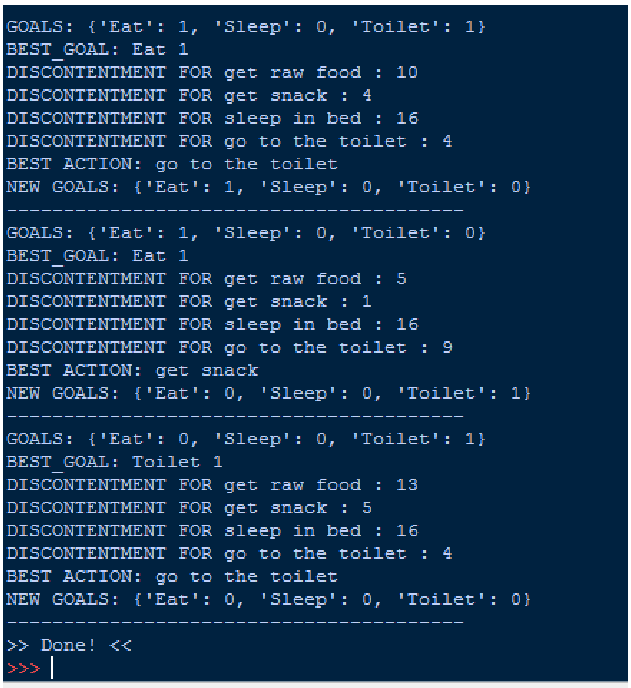
RESULTS OF CODE A:

Code A shows the example where eating food makes the AI feel the need to go to the toilet, after fulfilling its hunger needs. This creates a positive and a negative impact to separate goals. In the book "*Artificial Intelligence for Games*" by Ian Millington and John Fung. *Chapter 5, p.406,* it is stated that a weakness to this approach is that the AI "fails to take account of side effects than an action may have" causing erratic and un-humanlike behaviour.

This AI in this example begins with EAT and TOILET at the same value yet chooses to go to eat to fulfil that need by -3, whilst the toilet goal increases by 4. A human could determine that maybe going to the toilet first may avoid an embarrassing event occurring, or even a fatal event if used in a game where the AI's life was at stake using this decision making.



```
ACTIONS:
 * [get raw food]: {'Eat': -2, 'Toilet': 2}
 * [get snack]: {'Eat': -1, 'Toilet': 1}
 * [sleep in bed]: {'Sleep': -4}
 * [go to the toilet]: {'Toilet': -3}
>> Start <<
----------------------------------------
GOALS: {'Eat': 4, 'Sleep': 3, 'Toilet': 2}
BEST_GOAL: Eat 4
DISCONTENTMENT FOR get raw food : 20
DISCONTENTMENT FOR get snack : 18
DISCONTENTMENT FOR sleep in bed : 1
DISCONTENTMENT FOR go to the toilet : 1
BEST ACTION: go to the toilet
NEW GOALS: {'Eat': 4, 'Sleep': 3, 'Toilet': 0}
----------------------------------------
GOALS: {'Eat': 4, 'Sleep': 3, 'Toilet': 0}
BEST_GOAL: Eat 4
DISCONTENTMENT FOR get raw food : 8
DISCONTENTMENT FOR get snack : 10
DISCONTENTMENT FOR sleep in bed : 1
DISCONTENTMENT FOR go to the toilet : 9
BEST ACTION: sleep in bed
NEW GOALS: {'Eat': 4, 'Sleep': 0, 'Toilet': 0}
----------------------------------------
GOALS: {'Eat': 4, 'Sleep': 0, 'Toilet': 0}
BEST_GOAL: Eat 4
DISCONTENTMENT FOR get raw food : 8
DISCONTENTMENT FOR get snack : 10
DISCONTENTMENT FOR sleep in bed : 16
DISCONTENTMENT FOR go to the toilet : 9
BEST ACTION: get raw food
NEW GOALS: {'Eat': 2, 'Sleep': 0, 'Toilet': 2}
----------------------------------------
GOALS: {'Eat': 2, 'Sleep': 0, 'Toilet': 2}
BEST_GOAL: Eat 2
DISCONTENTMENT FOR get raw food : 16
DISCONTENTMENT FOR get snack : 10
DISCONTENTMENT FOR sleep in bed : 16
DISCONTENTMENT FOR go to the toilet : 1
BEST ACTION: go to the toilet
NEW GOALS: {'Eat': 2, 'Sleep': 0, 'Toilet': 0}
----------------------------------------
GOALS: {'Eat': 2, 'Sleep': 0, 'Toilet': 0}
BEST_GOAL: Eat 2
DISCONTENTMENT FOR get raw food : 4
DISCONTENTMENT FOR get snack : 2
DISCONTENTMENT FOR sleep in bed : 16
DISCONTENTMENT FOR go to the toilet : 9
BEST ACTION: get snack
NEW GOALS: {'Eat': 1, 'Sleep': 0, 'Toilet': 1}
----------------------------------------
```



```
GOALS: {'Eat': 1, 'Sleep': 0, 'Toilet': 1}
BEST_GOAL: Eat 1
DISCONTENTMENT FOR get raw food : 10
DISCONTENTMENT FOR get snack : 4
DISCONTENTMENT FOR sleep in bed : 16
DISCONTENTMENT FOR go to the toilet : 4
BEST ACTION: go to the toilet
NEW GOALS: {'Eat': 1, 'Sleep': 0, 'Toilet': 0}
----------------------------------------
GOALS: {'Eat': 1, 'Sleep': 0, 'Toilet': 0}
BEST_GOAL: Eat 1
DISCONTENTMENT FOR get raw food : 5
DISCONTENTMENT FOR get snack : 1
DISCONTENTMENT FOR sleep in bed : 16
DISCONTENTMENT FOR go to the toilet : 9
BEST ACTION: get snack
NEW GOALS: {'Eat': 0, 'Sleep': 0, 'Toilet': 1}
----------------------------------------
GOALS: {'Eat': 0, 'Sleep': 0, 'Toilet': 1}
BEST_GOAL: Toilet 1
DISCONTENTMENT FOR get raw food : 13
DISCONTENTMENT FOR get snack : 5
DISCONTENTMENT FOR sleep in bed : 16
DISCONTENTMENT FOR go to the toilet : 4
BEST ACTION: go to the toilet
NEW GOALS: {'Eat': 0, 'Sleep': 0, 'Toilet': 0}
----------------------------------------
>> Done! <<
>>>
```

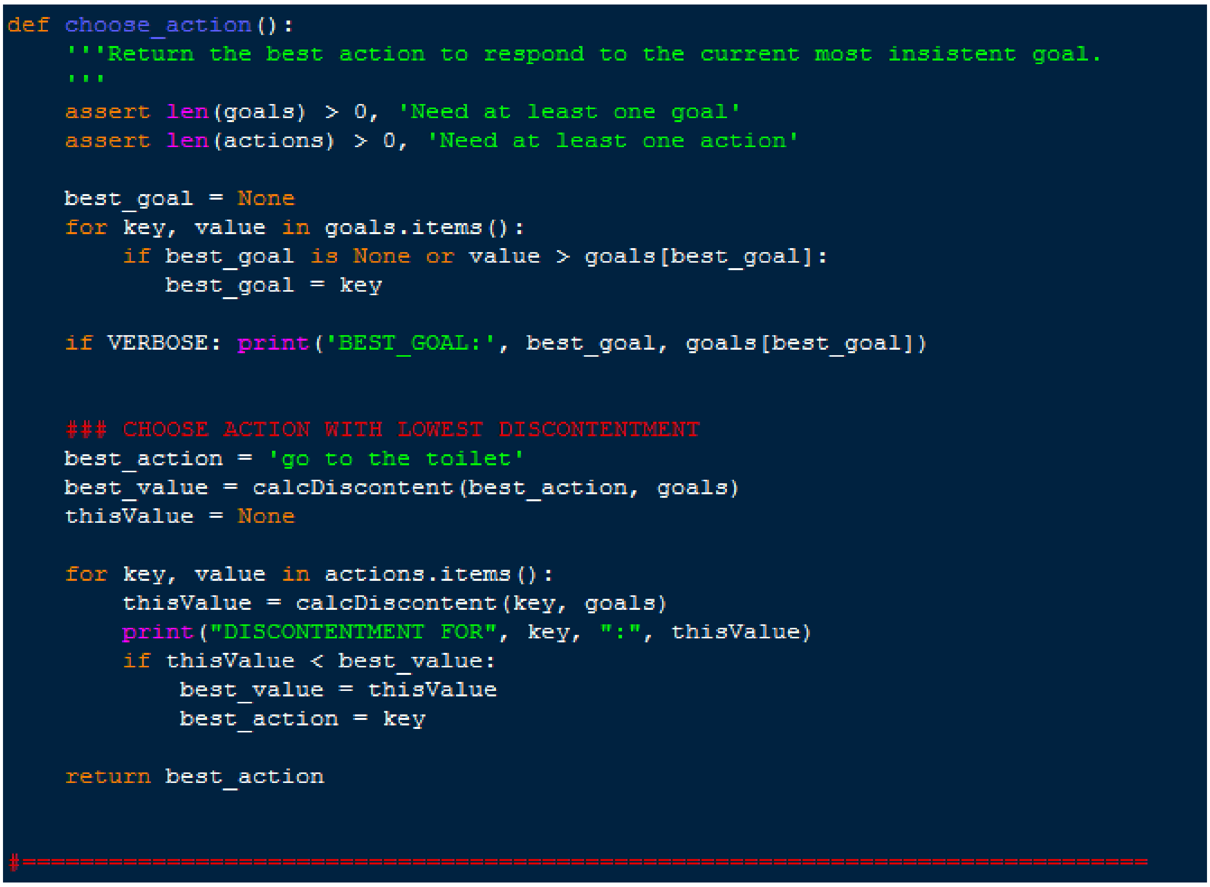
RESULTS OF CODE B:

Code B shows an example of using SGI with the determining factor being “discontentment”. Discontentment is calculated by squaring and multiplying the result of an action taken on an effected goal. If the action effects two goals (positively or negatively), then it is simply added to calculate the total discontentment of taking that action.

As you can see in the example of the first decision, the AI shows more human like behaviour, deciding to go to the toilet instead of adding to it’s discontentment.

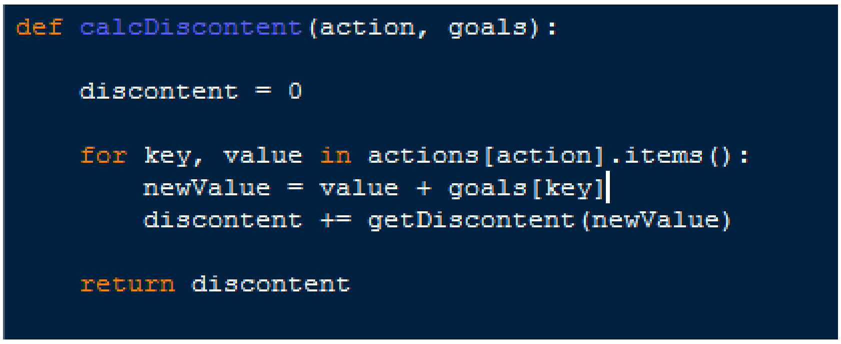
Code B:

ChooseAction:

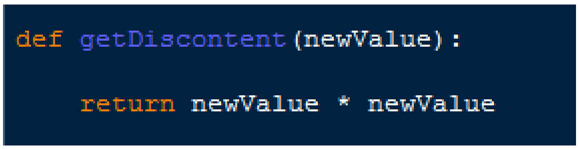


```
def choose_action():
    '''Return the best action to respond to the current most insistent goal.
    '''
    assert len(goals) > 0, 'Need at least one goal'
    assert len(actions) > 0, 'Need at least one action'

    best_goal = None
    for key, value in goals.items():
        if best_goal is None or value > goals[best_goal]:
           best_goal = key

    if VERBOSE: print('BEST_GOAL:', best_goal, goals[best_goal])


    ### CHOOSE ACTION WITH LOWEST DISCONTENTMENT
    best_action = 'go to the toilet'
    best_value = calcDiscontent(best_action, goals)
    thisValue = None

    for key, value in actions.items():
        thisValue = calcDiscontent(key, goals)
        print("DISCONTENTMENT FOR", key, ":", thisValue)
        if thisValue < best_value:
            best_value = thisValue
            best_action = key

    return best_action


#==============================================================================
```

CalculateDiscontentment:



```
def calcDiscontent(action, goals):

    discontent = 0

    for key, value in actions[action].items():
        newValue = value + goals[key]
        discontent += getDiscontent(newValue)

    return discontent
```



```
def getDiscontent(newValue):

    return newValue * newValue
```