Corey Crooks
ccrooks6@gatech.edu
Project 2 Reflection
GtID: 903130397

What is your agent's reasoning approach - visual or verbal? Does your agent need to convert between one or the other?

The agent's reasoning approach was the verbal approach. I continued to use this reasoning from Project 1 and will use the visual approach for the final project. Verbal reasoning seemed like it would be much easier to implement because I had very little experience programming in Python before starting Project 1. After becoming more familiar with Python, I feel more capable of using different libraries such as Pillow.

```
2x2
true
true
A
        a
                shape:square
                fill:yes
                size:very large
B
        b
                shape:square
                fill:yes
                size:very large
C
        c
                shape:square
                fill:yes
                size:very large
```

How does your agent represent/store the images efficiently? What is your agent's overall problem-solving process?

Project 1:
The overall problem-solving process includes both semantic networks and generate and test. Semantic networks allow the object representations to be compared to each other. Once the representations are discovered, the agent loops through each possible solution (1-6). For each possible solution, the semantic network represents the comparison from C to D, where D is within the range (1-6). In order to find a solution, the agent takes the transformation relationships

from A to B and C to D. Each attribute has a weight attached within the semantic network to allow for a more precise solution. The answer is finally chosen based on the highest similarity score from C to D (1-6). The images are stored by looping through all of the objects and adding them to a dictionary. Along with the objects, I had created a transformation dictionary to hold all of the transformations as well. Once these were set up, I figured out the comparison of shapes and attributes across figures A, B and C. I believed that having dictionaries to hold and store image properties was the best way to go.

Project 2:
The approach was pretty much the same as Project 1's approach by first representing all of the objects in the matrix and then applying Generate and Test. The agent stores and represents the images through two dictionaries for the transformations and their similarities.

Provide an overview of the design of your agent

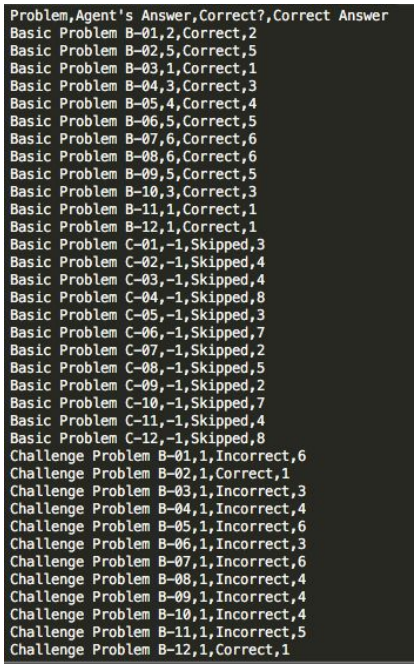
Project 1:
As described above, the overview starts with the basics of semantic networks. The agent reads in each objects attribute within each figure and compares them to each other. The comparisons are then recorded and used to find the closest similarity between C and D (1-6). The main classes provided within this project are Agent.py, SemanticNetwork.py and Weights.py. The agents similarities dictionary is built by getting the comparisons between A to B and A to C then goes through each possible solution 1-6. Once the similarity dictionary is completed, the agent can return the figure with the highest similarity score. The design of the agent came was build off of a one problem at a time approach while looking at the harder solutions to see if the agent would be able to solve those as well. By being able to take one problem at a time, the semantic network was able to grow bigger by making more distinct comparisons horizontally and vertically.

Project 2:
Project 2 has the same design as Project 1 however it adds a few different layers of complexity to handle 3x3 transformations. The agent follows suite with the original concept of building the semantic network and using generate and test. The semantic network is built by using horizontal, vertical and diagonal transformations and generate and test is used to test all of the possible answers for the closest similarity score. I use similarity scoring because when using generate and test some of the answers are very close and needed pick the best match.

What mistakes does your agent makes? Could these mistakes be resolved within your agent's current approach, or are they fundamental problems with the way your agent approaches these problems?

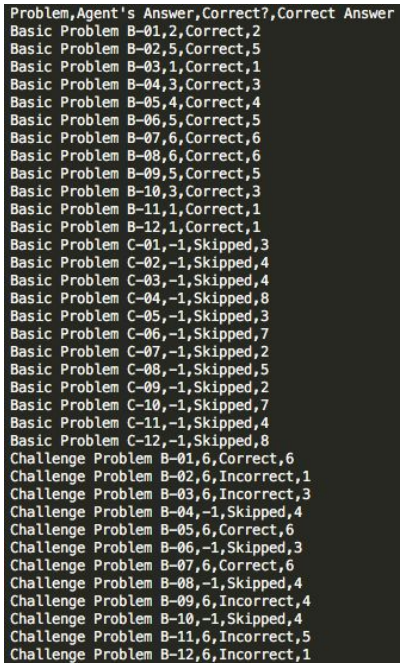
Project 1:

The agent within my project makes a mistakes by generating a random tie breaker which proceeds to answer any non handled problem with special attributes by selecting a random answer for it. The result of this is simply all of the challenge problems answering differently upon each run since they were not handled. These problems can be resolved with the current approach by simply making the network smarter and score attributes differently however I would like to use a more hybrid approach when trying to solve the more challenging problems. When using a hybrid approach scores can be more distinctive and even be gauged by using an x and y location plane. The semantic network would be more fine tuned for solving much more difficult problems.



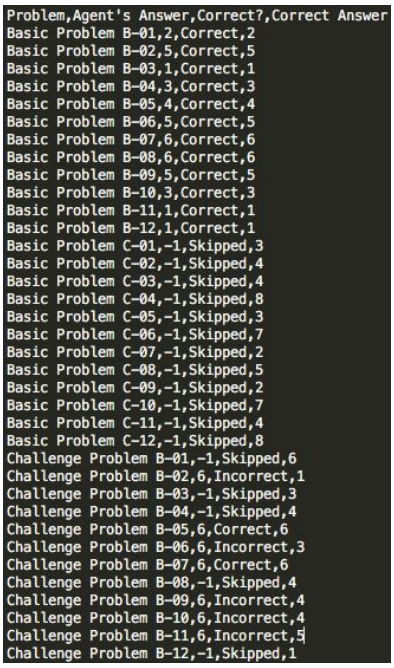
```
Problem,Agent's Answer,Correct?,Correct Answer
Basic Problem B-01,2,Correct,2
Basic Problem B-02,5,Correct,5
Basic Problem B-03,1,Correct,1
Basic Problem B-04,3,Correct,3
Basic Problem B-05,4,Correct,4
Basic Problem B-06,5,Correct,5
Basic Problem B-07,6,Correct,6
Basic Problem B-08,6,Correct,6
Basic Problem B-09,5,Correct,5
Basic Problem B-10,3,Correct,3
Basic Problem B-11,1,Correct,1
Basic Problem B-12,1,Correct,1
Basic Problem C-01,-1,Skipped,3
Basic Problem C-02,-1,Skipped,4
Basic Problem C-03,-1,Skipped,4
Basic Problem C-04,-1,Skipped,8
Basic Problem C-05,-1,Skipped,3
Basic Problem C-06,-1,Skipped,7
Basic Problem C-07,-1,Skipped,2
Basic Problem C-08,-1,Skipped,5
Basic Problem C-09,-1,Skipped,2
Basic Problem C-10,-1,Skipped,7
Basic Problem C-11,-1,Skipped,4
Basic Problem C-12,-1,Skipped,8
Challenge Problem B-01,1,Incorrect,6
Challenge Problem B-02,1,Correct,1
Challenge Problem B-03,1,Incorrect,3
Challenge Problem B-04,1,Incorrect,4
Challenge Problem B-05,1,Incorrect,6
Challenge Problem B-06,1,Incorrect,3
Challenge Problem B-07,1,Incorrect,6
Challenge Problem B-08,1,Incorrect,4
Challenge Problem B-09,1,Incorrect,4
Challenge Problem B-10,1,Incorrect,4
Challenge Problem B-11,1,Incorrect,5
Challenge Problem B-12,1,Correct,1
```

Run 1



```
Problem,Agent's Answer,Correct?,Correct Answer
Basic Problem B-01,2,Correct,2
Basic Problem B-02,5,Correct,5
Basic Problem B-03,1,Correct,1
Basic Problem B-04,3,Correct,3
Basic Problem B-05,4,Correct,4
Basic Problem B-06,5,Correct,5
Basic Problem B-07,6,Correct,6
Basic Problem B-08,6,Correct,6
Basic Problem B-09,5,Correct,5
Basic Problem B-10,3,Correct,3
Basic Problem B-11,1,Correct,1
Basic Problem B-12,1,Correct,1
Basic Problem C-01,-1,Skipped,3
Basic Problem C-02,-1,Skipped,4
Basic Problem C-03,-1,Skipped,4
Basic Problem C-04,-1,Skipped,8
Basic Problem C-05,-1,Skipped,3
Basic Problem C-06,-1,Skipped,7
Basic Problem C-07,-1,Skipped,2
Basic Problem C-08,-1,Skipped,5
Basic Problem C-09,-1,Skipped,2
Basic Problem C-10,-1,Skipped,7
Basic Problem C-11,-1,Skipped,4
Basic Problem C-12,-1,Skipped,8
Challenge Problem B-01,6,Correct,6
Challenge Problem B-02,6,Incorrect,1
Challenge Problem B-03,6,Incorrect,3
Challenge Problem B-04,-1,Skipped,4
Challenge Problem B-05,6,Correct,6
Challenge Problem B-06,-1,Skipped,3
Challenge Problem B-07,6,Correct,6
Challenge Problem B-08,-1,Skipped,4
Challenge Problem B-09,6,Incorrect,4
Challenge Problem B-10,-1,Skipped,4
Challenge Problem B-11,6,Incorrect,5
Challenge Problem B-12,6,Incorrect,1
```

Run 2



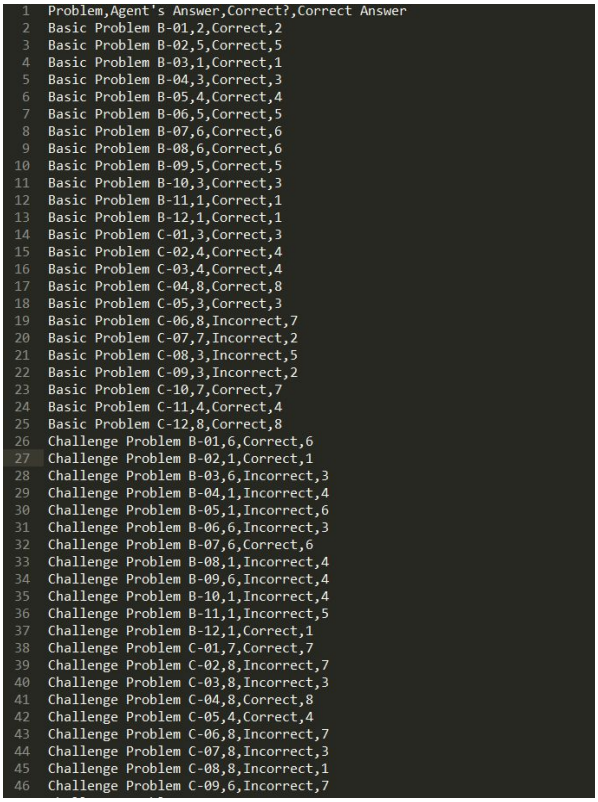
```
Problem,Agent's Answer,Correct?,Correct Answer
Basic Problem B-01,2,Correct,2
Basic Problem B-02,5,Correct,5
Basic Problem B-03,1,Correct,1
Basic Problem B-04,3,Correct,3
Basic Problem B-05,4,Correct,4
Basic Problem B-06,5,Correct,5
Basic Problem B-07,6,Correct,6
Basic Problem B-08,6,Correct,6
Basic Problem B-09,5,Correct,5
Basic Problem B-10,3,Correct,3
Basic Problem B-11,1,Correct,1
Basic Problem B-12,1,Correct,1
Basic Problem C-01,-1,Skipped,3
Basic Problem C-02,-1,Skipped,4
Basic Problem C-03,-1,Skipped,4
Basic Problem C-04,-1,Skipped,8
Basic Problem C-05,-1,Skipped,3
Basic Problem C-06,-1,Skipped,7
Basic Problem C-07,-1,Skipped,2
Basic Problem C-08,-1,Skipped,5
Basic Problem C-09,-1,Skipped,2
Basic Problem C-10,-1,Skipped,7
Basic Problem C-11,-1,Skipped,4
Basic Problem C-12,-1,Skipped,8
Challenge Problem B-01,-1,Skipped,6
Challenge Problem B-02,6,Incorrect,1
Challenge Problem B-03,-1,Skipped,3
Challenge Problem B-04,-1,Skipped,4
Challenge Problem B-05,6,Correct,6
Challenge Problem B-06,6,Incorrect,3
Challenge Problem B-07,6,Correct,6
Challenge Problem B-08,-1,Skipped,4
Challenge Problem B-09,6,Incorrect,4
Challenge Problem B-10,6,Incorrect,4
Challenge Problem B-11,6,Incorrect,5
Challenge Problem B-12,-1,Skipped,1
```

Run 3

Project 2:

The agent has been corrected to handle the mistakes that were happening in project 1 however does contain a few mistakes introduced by problem set c. The agent struggles solving the following problems: C7, C8, C9, C10, C11 and C12. The reason why the agent struggles is because of the similarity score result in which it is coming very close to other similar figures. The problem also struggles with figures that contain any overlapping features which doesn't

quite handle them well. These mistakes could essentially be fixed with the approach my agent takes by figuring out a more efficient way of scoring the results of the semantic network and also figuring out how to handle specific cases of overlapping.



```
Problem,Agent's Answer,Correct?,Correct Answer
Basic Problem B-01,2,Correct,2
Basic Problem B-02,5,Correct,5
Basic Problem B-03,1,Correct,1
Basic Problem B-04,3,Correct,3
Basic Problem B-05,4,Correct,4
Basic Problem B-06,5,Correct,5
Basic Problem B-07,6,Correct,6
Basic Problem B-08,6,Correct,6
Basic Problem B-09,5,Correct,5
Basic Problem B-10,3,Correct,3
Basic Problem B-11,1,Correct,1
Basic Problem B-12,1,Correct,1
Basic Problem C-01,3,Correct,3
Basic Problem C-02,4,Correct,4
Basic Problem C-03,4,Correct,4
Basic Problem C-04,8,Correct,8
Basic Problem C-05,3,Correct,3
Basic Problem C-06,8,Incorrect,7
Basic Problem C-07,7,Incorrect,2
Basic Problem C-08,3,Incorrect,5
Basic Problem C-09,3,Incorrect,2
Basic Problem C-10,7,Correct,7
Basic Problem C-11,4,Correct,4
Basic Problem C-12,8,Correct,8
Challenge Problem B-01,6,Correct,6
Challenge Problem B-02,1,Correct,1
Challenge Problem B-03,6,Incorrect,3
Challenge Problem B-04,1,Incorrect,4
Challenge Problem B-05,1,Incorrect,6
Challenge Problem B-06,6,Incorrect,3
Challenge Problem B-07,6,Correct,6
Challenge Problem B-08,1,Incorrect,4
Challenge Problem B-09,6,Incorrect,4
Challenge Problem B-10,1,Incorrect,4
Challenge Problem B-11,1,Incorrect,5
Challenge Problem B-12,1,Correct,1
Challenge Problem C-01,7,Correct,7
Challenge Problem C-02,8,Incorrect,7
Challenge Problem C-03,8,Incorrect,3
Challenge Problem C-04,8,Correct,8
Challenge Problem C-05,4,Correct,4
Challenge Problem C-06,8,Incorrect,7
Challenge Problem C-07,8,Incorrect,3
Challenge Problem C-08,8,Incorrect,1
Challenge Problem C-09,6,Incorrect,7
```

Please detail on your evaluation/performance criterias and your agent's results. Think about accuracy, efficiency and generality. Are there other metrics or scenarios under which you think your agent's performance would improve or suffer?

Project 1:

The agents results for this first project were positive. The agent handles all of the given B problems and solves them by measuring the different transformations along with with their scores to find the appropriate answer. I started to build the system by taking a one problem at a time approach while keeping the end goal (solving harder problems) in mind. The way that the agent current evaluates the transformations by applying specific weights are efficient enough to handle the first set of problems, however as the problems get more challenging so does the

scoring system. For the second project, handling the 3x3 matrices is really going to put the current system to the test. Improvisation will have to be implied to handle the extra set of matrices, along with more comparisons and transformations as things become more complex.

Project 2:
First and foremost, the agent performs much better than the original Project 1 by handling transformations better. The agent is able to solve all of the problems from the given set B and approximately 8 out of 12 of the problem set C. I believe that if the agent is asked to solve similar problems then the agent would behave the same and solve approximately the same amount of problems. The agent would fail for any conditions outside of the scope has not been accounted for. After building these two projects, I can see the powerfulness of adapting learning by recorded cases because the agent would be able to solve new problems based on previously seen problems.

Please provide an explanation of how your methods/components/ideas in your agent's design are/might be similar to (or can be related) to specific KBAI methods discussed in class

Project 1:
The methods that were used to get a working solution of this project came 100% from lectures. From building a semantic network to generate and test, these methods allowed me to build a system smart enough to handle basic problems. When first designing the system, I tried to mimic the semantic network horizontal and vertical transformations that were seen in class for Figure A to Figure B and then repeated it for every possible answer. I had then created a class called Weights.py to handle the weights of similarity for each transformation. This allowed to build a solution that would possible fit the problem. As project complexity increases the semantic network will have to become much stronger to handle different transformations that the agent has not seen before. Along with these transformations, it is going to have to handle them over a 3x3 plane. After the agent is able to handle those for the final project the agent will have to visually build the semantic network while still using generate and test to generate a possible similarity score then testing each transformation from C to 1-6.

Project 2:
The same description applies to project 2 as in project 1, where the entire project's solution came specifically from each individual class lecture. The entire structure of the project remained fundamentally intact and continued to use Semantic Networks and Generate and Test. As an answer to a previously asked question, I suggested that the usage of learning by recording test

cases could also be applied for future renditions of this project to dynamically solve problems the agent has never seen before.

What does the design and performance of your agent tell us about human cognition? How is it similar, and how is it different? Has your agent's performance given you any insights into the way people solve these problems?

Overall the design and performance tells us a lot about human cognition. It shows that the mind is very complex and continuously makes many decisions, interpretations and computations without us even being aware of it. While us humans can solve many Raven's Matrices problems almost instantaneously, an agent needs to break down every single transformation into smaller steps before attempting to solve the problem. This shows us that both humans and artificial intelligence agents think completely differently. However my agent tries to solve problems as I would solve the problem by looking at the changes from each of the figures then trying to find an answer that is closest in relation. The agent gives insight by handling the weights internally where as us humans use similar reasoning for picking out the best answer.