Sai Srujana Buddi (GT ID: 903141698)

Project1 Reflection: ADDRESSING RAVEN’S PROGESSIVE MATRICES USING SEMANTIC NETWORKS

1. INTRODUCTION:

This report addresses how an AI agent can solve Raven’s progression matrices. I employed semanctic networks approach to solve RPM for Project 1. To reduce computational complexity I am using verbal files for the project

2. SOLVING RPM USING SEMANTIC NETWORKS

Algorithm:

The main **solve** function:

1. Read all the figures available

2. Find relation between lexicons of A and B i.e., to build a vocabulary to describe relation between lexicons and structurally represent the relation between A and B

3. Semantical identification and representation

4. Relate lexicons A and C

5. Built the relations and semantic networks between C and each option for D provided.

6. Built the realtions and semantic networks between B and each option provided for D

7.Compare semantic network of A:B with C:{1,2,3,4,5,6}

8. Assign score for {1,2,3,4,5,6} depending on the match of C:{option} with A:B

9. Comapre semantic network of A:C with B:{1,2,3,4,5,6}

10. Assign score for {1,2,3,4,5,6} depending on the match of B:{option} with A:C

11. The option with maximum score is selected as the answer

The **FindCorrespondingObjects** function:

This function maps the objects between two images

1. Sort all the objects in A and B

2. For every object in A and B list all the attributes and save them in a dictionary

3. If number of objects in A is less than or equal to the number of objects in B then for every object in A I map some object in B depending on the similarity index between the objects.

4. The similarity index is calculated as follows: If two objects have same numberof attributes then similarity index between these two objects is increased by one. If any of the attributes of the two objects have same value then the similarity index is increased by one. Finally for an object in A the object in B with maximum similarity index is mapped as the corresponding object.

5. If number of objects in A is less than number of objects in B, then all the objects in B to which none of the objects in A are mapped are indexed as ‘added’

6. Else if number of objects in B is more than number of objects in A, then for every object in B I map an object in A which has the highest similarity index with the object in B

7. If number of objects in A is more than number of objects in B, then all the objects in A to which none of the objects in B are mapped are indexed as ‘deleted’

8. Dictionary of corresponding elements is returned from the function

The **GetRelations** function:

This function build the semantic network i.e., it describes the relational changesoccuring between the two images.

1. The dictionary returned from the FindCorrespondingObjects function is used as input to this function

2. For every key (object of A) in the dictionary, there exists a value(object of B). When the indices are not equal to ‘added’ or ‘deleted’ then the attributes of both the objects are compared and a relation is drawn

3. This is repeated for all the objects in fiugres A and B

4. The semantic network representation I am employing is in the form of a dictionary that stores the values of objects, the attributes of the objects and relative changes in the attribute value between figure A and B

5. This function returns a dictionary in the format mentioned above

The **CompareRelations** Function:

This function is used to compare the relational/semantic changes between AtoB and CtoFig{1,2,3,4,5,6}

1. The dictionary of relations which is returned from the above GetRelations function is used as input to this function.

2. This function compares the two semantic networks and gives a similarity score.

3. The similarity score is computed as: for every object’s attribute in semantic relation1, if it matches with the corresponding object’s attribute in semantic relation2 then the similarity score is increased by 1.

4. The score is returned to the Solve function where the object with maximum score is selected as answer

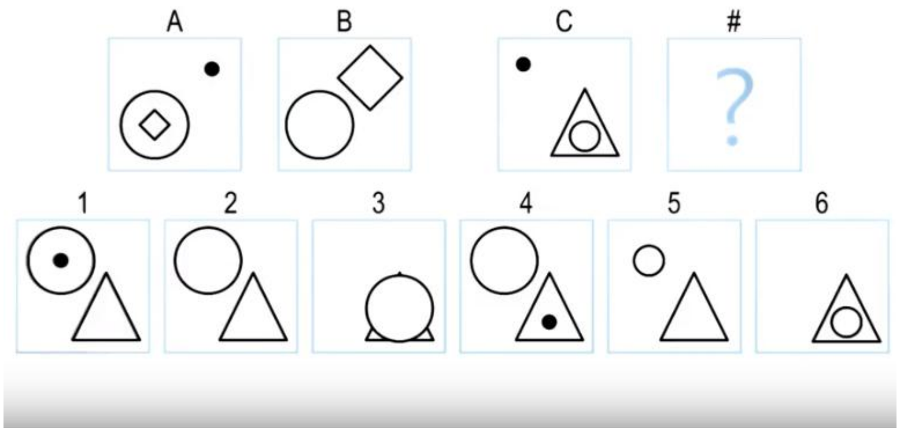

Figure 1. RPM with a complex semantic networks

The way in which the AI agent I designed would solve a RPM problem, RPM in Fig 1 for this instance is as below:

Find corresponding objects:

A	B
A.Y.shape:rhombus	B.Y.shape:rhombus
A.X.shape=circle	B.X.shape=circle
A.Y.X=inside	B.Y.X=above
A.Z=circle	B.Z=deleted
A.Z.fill=yes	B.Z.fill=na
A.Y.size=small	B.Y.size=big
A.Z.X=above	B.Z.X=na

Table 1

Get relations and compare the semantic changes:

Comparison sheet of A and B		Comparison sheet of C and 2	Score
A.Y.shape:rhombus		C.Y.shape:circle	0
A.X.shape=circle		C.X.shape=triangle	0
A.Y.X=inside	==	C.Y.X=inside	1
A.Z=circle	==	C.Z=circle	1
A.Z.fill=yes	==	C.Z.fill=yes	1
A.Y.size=small	==	C.Y.size=small	1
A.Z.X=above	==	C.Z.X=above	1
A.X.Y.position=left		C.X.Y.position=right	0
A.Z.position=right		C.Z.position=left	0

B.Y.shape:rhombus		6.Y.shape:rhombus	0
B.X.shape=circle		6.X.shape=circle	0
B.Y.X=above	==	6.Y.X=above	1
B.Z=deleted	==	6.Z=deleted	1
B.Z.fill=na	==	6.Z.fill=na	1
B.Y.size=big	==	6.Y.size=big	1
B.Z.X=na	==	6.Z.X=na	1
			10

Table 2 Comparison sheets between (A,B) and (C,2)

In final stage, the tester ranks all scores from the previous stage as shown in the Table 4 and takes the highest one. In this question the #2 had the highest score of 10. So it is the derived solution.

Option	Score
1	7
2	10
3	6
4	7
5	9
6	8

Table 3. Scores of options provided

Areas of Improvement:

1. Figures or visual data is to be used instead of the verbal data.

2. My AI agent is entirely working on basis of semantic networks which is performing efficiently for the given cases but it might not work for more sophisticated problems. Employing generate and test method and deriving relations and semantic networks based on the scenario can be more efficient.

3. I am not defining a weight function i'e., while calculating the similarity index/ score I am assigning a score of 1 for all sorts of changes. This might make my system unbiased. Instead I can bias the system by using weight functions for changes as shown in the table below

Relation	Weight
Unchanged	5
Reflected	4

Rotated	3
Scaled	2
Deleted	1
Shape unchaged	0

Table 1

4. My FindCorrespondingObjects function is not working as per expectation in the cases where number of objects in first image is more than number of objects in second image. (This might be teh reason why my AI agent is failing for Basic Problem B-12) I have to design a more robust method for mapping the objects between two figures

If we are to compare the AI agent's reasoning to human cognition, like human the agent uses observations and forms base conclusions from these observations. It then augments these conclusions based on other tests, just like we do when we look at these problems. These models can be time consuming as semantic networks are to be drawn using each and every object. I believe, the effectiveness of the model can be increased by embedding learning and meta cognition in the AI agents.