Project 2, CS7637

Shantanu Singh, shant0602@gatech.edu

Project Approach

Following were the different stages of development of the AI agent implemented for RPM solution.

1. Extension of Affine transformation for 3x3 matrices

- I attempted to extend the approach of affine transformation used in first project further to 3x3 matrices.
- However, unlike 2x2 matrix where the possible combinations were only A:B::C:? and A:C::B:?; there were several combinations possible in 3x3 matrix [1].
- **Figure 1** and **Table 1** show the possible combinations in 3x3 matrices.
- **Table 2** shows the unary transformations for doublet pairs and binary transformations for triplets [2].

- Each transformation was recorded with a similarity value calculated by means of Tversky's ratio [3] calculated as shown below.

$$similarity(A, B) = \frac{f(A \cap B)}{f(A \cup B)}$$

- The transformation with the highest similarity value was used as an analogy to generate the unknown matrix; which was then matched against the possible answer choices to get the correct answer.
- However, this approach proved to be extremely time intensive and the rate of success was low: 5/12 in Basic problem C and 2/12 in Challenge problem C.
- Ignoring the doublets comparison reduced the time taken, but this approach had to be abandoned due to its low success rate.

2. Implementation using Dark Pixel Ratio

- In the previous approach; transformations were sorted by their similarity ratio, which turned out to be very time intensive.

- I felt the need to switch to an alternative way of comparing images which could straight away decide the transformation rather than comparing all the possible transformations first and then deciding the one.
- Images were compared by means of Dark Pixel defined as:
  'The difference in percentage of the number of dark-colored pixels with respect to the total number of pixels in the contiguous pixel sets of two matrix cells' [4].
- The agent used a hierarchy of transformational methods starting from union followed by intersection, subtraction, XOR and pixel progression.
- Later, the concept of intersection pixel was also incorporated in the agent.
- The agent was tested for different problems and the methods were refined subsequently and threshold values were finalized.

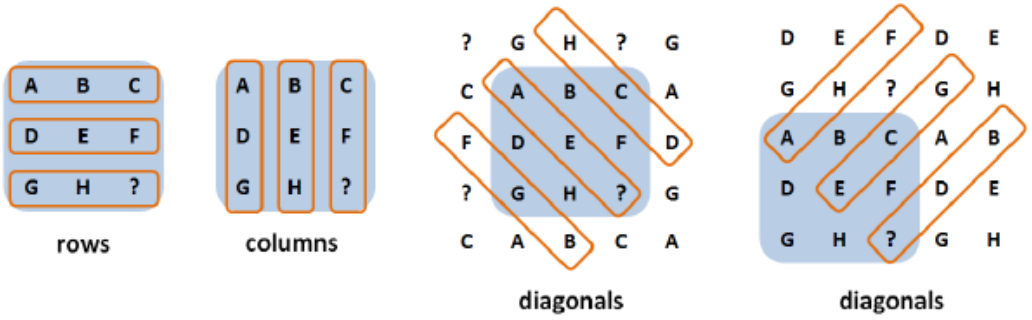

Figure 1: Collinear triplets and parallel set of collinear triplets in 3x3 matrix *[1]*

	Rows	Columns	Diagonals	
Pairs	A : B :: H : ? B : C :: H : ? D : E :: H : ? E : F :: H : ? G : H :: H : ? A : C :: G : ? D : F :: G : ?	A : D :: F : ? D : G :: F : ? B : E :: F : ? E : H :: F : ? C : F :: F : ? A : G :: C : ? B : H :: C : ?	F : G :: E : ? G : B :: E : ? H : C :: E : ? C : D :: E : ? A : E :: E : ? F : B :: A : ? H : D :: A : ?	F : H :: D : ? H : A :: D : ? G : C :: D : ? C : E :: D : ? B : D :: D : ? F : A :: B : ? G : E :: B : ?
Triplets	A : B : C :: G : H : ? D : E : F :: G : H : ?	A : D : G :: C : F : ? B : E : H :: C : F : ?	F : G : B :: A : E : ? H : C : D :: A : E : ?	F : H : A :: B : D : ? G : C : E :: B : D : ?

Table 1: List of combinations in a 3x3 matrix *[1]*

Transformation No.	Unary (Applied on A and the result compared with last element of doublet)	Binary (Applied on first 2 elements of triplet and the result is compared with last element of triplet)
1.	Identity	Union
2.	Rotate90	Intersection
3.	Rotate180	A-B
4.	Rotate270	B-A
5.	Identity-flip	XOR
6.	Rotate90-flip	
7.	Rotate180-flip	
8.	Rotate270-flip	

Table 2: Unary and Binary transformations

Final Agent Development

1. Assumptions for the Current Model Representations in Agent

- The representation of images are 2-D arrays of grayscale pixels. Thus, each pixel is associated with a single intensity value (0: white and 1: black). The code explicitly converts the supplied image to grayscale.
- While matching two images the agent displaces the image by 1 pixel in all directions to account for error tolerance in the position of pixels in image.

2. Working of the final agent

The agent implements visual approach to solve the RPM. Images are loaded from the dictionary of Raven figure objects.

Pseudocode of Agent's algorithm

STEP 1: Load each problem

STEP 2: Apply the following transformation in given order:

STEP 3: If transformation for two row triplets is UNION:

Generate the missing entry and get the correct answer by testing against each possible choice, GOTO STEP 9.

Repeat STEP 3 for two column triplets

STEP 4: If transformation for two row triplets is INTERSECTION:

Generate the missing entry and get the correct answer by testing against each possible choice, GOTO STEP 9

Repeat STEP 4 for two column triplets

STEP 5: If transformation for two row triplets is SUBTRACTION:

Generate the missing entry and get the correct answer by testing against each possible choice, GOTO STEP 9

Repeat STEP 5 for two column triplets

STEP 6: If transformation for two row triplets is XOR:

Generate the missing entry and get the correct answer by testing against each possible choice, GOTO STEP 9

Repeat STEP 6 for two column triplets

STEP 7: If transformation for two row triplets is PIXEL PROGRESSION:

Get the correct answer by testing the pixel difference against each possible choice,
GOTO STEP 9

Repeat STEP 7 for two column triplets

STEP 8: If no transformation matches return -1.

STEP 9: Repeat from STEP1, until all problems are loaded.

- Both the triplets of row or column as shown in **Table 1** are checked for the transformation patterns to ascertain strong relationship.
- The dark pixel percentage difference threshold is taken to be 1%

Problems faced in the design and Performance of the agent

1. Problems faced in design

- Agent performance was 8/12 correct on Basic problems C and 2/12 correct on challenge problem C, with the above pseudocode approach.
- The basis on which pixel progression works is as follows:
  There is a pixel progression $P_A - P_B = P_B - P_C$ (P = no. of dark pixels). This method handles problems having shapes being added across a row/column to the starting matrix of a triplet as shown in **Figure 2**.
- However, a loop hole of pixel progression method was exposed in Challenge Problem–02 as shown in **Figure 3**
- In Challenge Problem-02' this method gave 6 as the correct answer whereas 7 is the correct answer. According to the logic of this method both 6 and 7 are correct because this method captures a shape being added but it does not capture the orientation of that shape. Therefore, the answer to be encountered first is selected as the correct answer.
- A method which could handle this anomaly was required. A new method 'Incremental Progression' was implemented. This method is based on comparing the rates of pixel change and the rates of intersection pixel change along a row/column. Essentially this method compares second order pixel change across the entries in a row/column.

  PSEUDOCODE of 'Incremental Progression'

  STEP 1: Loop over a row:

  STEP 2: Calculate the rate of dark pixel change across a row.

  STEP 3: Calculate the rate of intersection pixel change across a row.

  STEP 4: Repeat STEP 1, unless last row is reached.

STEP 5: Calculate the respective range of 1) rate of dark pixel change and 2) rate of intersection pixel change from the values obtained for two rows.

STEP 6: Loop over each answer choice:

STEP 7: With the answer choice as last element of last row, calculate the dark pixel change rate and intersection pixel change rate.

STEP 8: Calculate and store the deviation of the rates in STEP 7 from the range calculated in STEP 5 , in an array.

STEP 9: Repeat STEP 6, until no answer choice is left.

STEP 10: The answer choice having the least deviation and still above a threshold (determined by hit and trial) is the final answer.

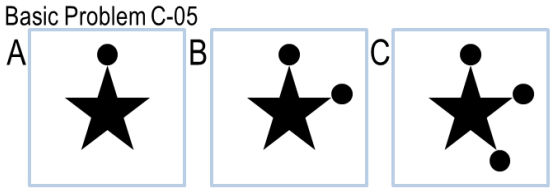

Figure 2: Basic problem 5

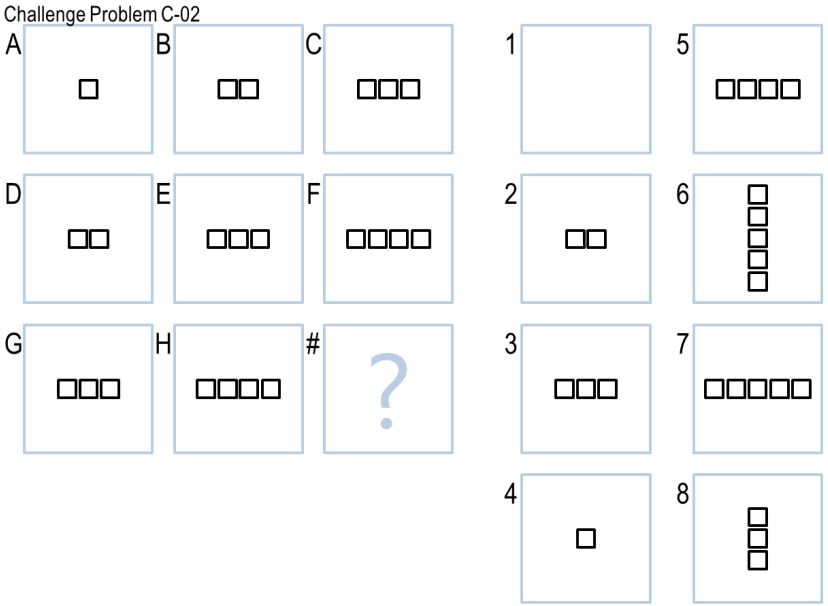

Figure 3: Challenge problem 2

2. Performance of the Agent

	Results of the final version of Agent	Progression of results over the course of development of agent
Basic C	11/12	5->5->5->8->11
Test C	10/12	7->6->7->9->10
Raven C	7/12	4->3->5->6->7
Challenge C	4/12	2->2->3->4->4

Table 3: Performance of the agent

A. Type of test cases failed by the agent and their solution

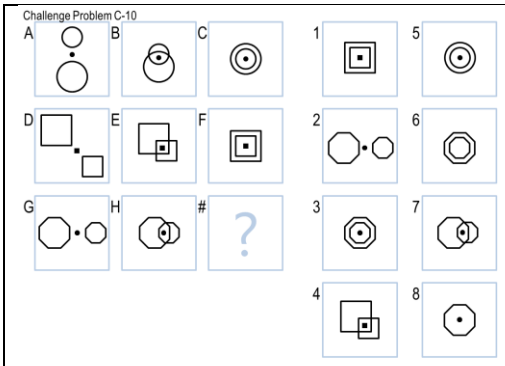	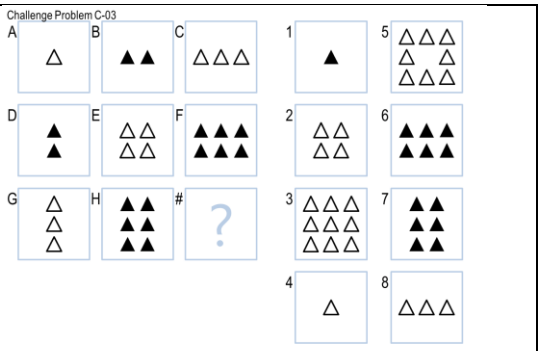
Problem: The agent can't read the spatial progression of objects as shown in above.	The agent fails to read the progression when fill is involved with the changing patterns.
Solution: Second order geometric analogy method is being tested to resolve this issue.	**Solution**: A way to resolve this issue must be thought of.

Table 4: Cases where the agent failed

B. How General is the Agent? Performance on Test and Basic Problem and Limitations

The agent is quite robust for problems which involve binary transformations and progression of similar type of shapes as were most of the problems in Basic C and Test C. The agent score 11/12 on Basic C and 10/12 on Test C which is a good score. However, the agent can't decipher patterns of progressions with different type of objects as discussed in the previous section. A methodology to rectify this situation is being developed now.

Shantanu Singh, shant0602@gatech.edu

Connection with Human Cognition

- Although the agent in consideration is essentially based upon a set of conditions written in form of an object-oriented program; still it has a weak connection with the process of human cognition. The agent is basically a semantic network. Different states are generated and tested for success to conclude the best path for the mapping.
- The agent has been revised multiple times by learning from its mistakes. It has been improved by incorporating new methods to tackle different type of RPM problems. This process is identical to the way humans learn from experience.
- Up to a brief extent, the problem-solving procedure of agent reflects the human problem-solving behavior in the sense that the agent matches the images step by step based on a sequence of various transformations. However, unlike humans the agent cannot cut short a mapping by recognizing that this path is leading to a wrong solution. The agent would go till the end and match the transformation with a similarity measure unlike a human who can abandon a path just by looking and predicting that path is not going to match.

Currently the agent is not really a true AI as its not learning from its past experiences to predict new transformations and relationship. Thus, it doesn't resemble the cognition of a human solving RPM in this aspect. However, the pattern of problem solving does resembles the approach of humans. Thus, it would be apt to say that the methodology used to solve the problem partly resembles to that of humans.

References

[1] M. Kunda, "VISUAL PROBLEM SOLVING IN AUTISM, PSYCHOMETRICS, AND AI: THECASE OF THE RAVEN'S PROGRESSIVE MATRICES INTELLIGENCE TEST," 2013.

[2] M. Kunda and A. K. G. Keith McGreggor, "A computational model for solving problems from the Raven's Progressive Matrices intelligence test using iconic visual representations," *Cognitive Systems Research,* no. Article in Press, 2012.

[3] A. Tversky, " Features of similarity," *Psychological Review, 84,* vol. 84, no. 4, pp. 327-352, 1977.

[4] D. A. Joyner, D. Bedwell, C. Graham, W. Lemmon, O. Martinez and A. K. Goel, "Using Human Computation to Acquire Novel Methods for Addressing Visual Analogy Problems on Intelligence Tests," in *International Conference on Computational Creativity*.