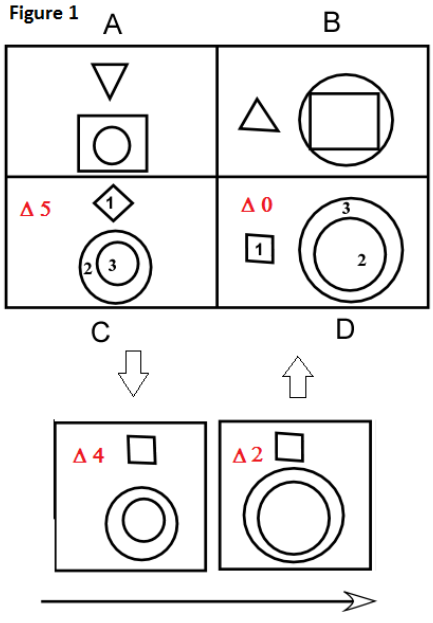
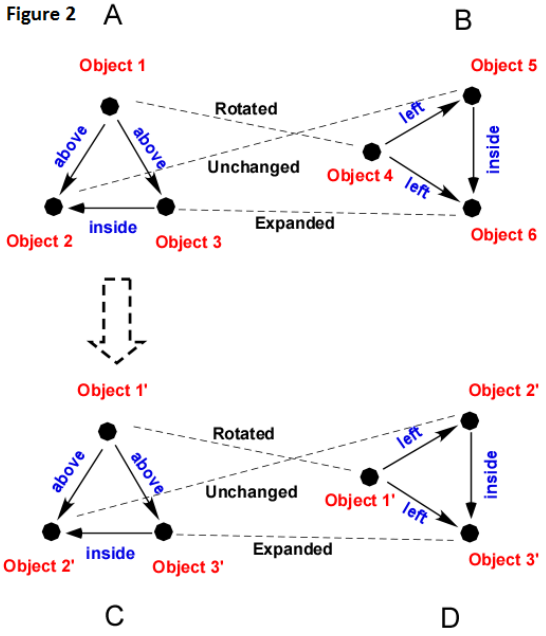
CS7637
Assignment 2

How to design an agent to address Raven's Progressive Matrices?

Prologue:

Computers are very powerful, and since the first day there were invented, they already changed our world. Computers can do millions calculation in a second, which humans may take years. However, in some situations, a baby can easily beat a super computer. For example, a baby can easily identify his/her parents among a group of strangers, while using computers for facial recognition is still a challenge to many engineers. The reason is human babies have a magic power called "human intelligence", which computers do not have yet. To examine human intelligence, many tests were developed, and among them including the famous Raven's progressive matrices (RPM). RPM is a nonverbal visual test, and since it is visual, it can be used on children at low age (< 5 years old). During the test, the human examinee is provided three sets of patterns of shapes (Figure 1 A, B and C) with a list of candidate patterns of shapes (not included in Figure 1), and depending on the difference between A and B, asked to figure out the match solution (Figure 1D) from the candidate list.





The challenges and potential solutions:

The challenges for designing an agent to solve RPM partly due to the inaccuracy of RPM itself. What's the exact meaning of difference between A and B, and how to describe the difference in a quantitative way so that the computers can understand and further compute? In my previous assignment, the use of **Semantic network** to solve RPM has been discussed. As shown in Figure 2, patterns of shapes are isolated as objects, the positions of these objects are identified, and finally the difference between Figure 1 A and B, in other words, the relationship between states is quantitatively described by a scored operation function. In my previous assignment, I also slightly touched on a strategy, **Generate & test**, to solve RPM, which used the information gained through semantic network to generate the ideal answer, and compared with all candidate solutions to identify the right solution. The details has not yet been discussed.

How to use Means-Ends Analysis in RPM :

Here, I will further discuss how to use the technique **Means-Ends Analysis** to generate the ideal answer for RPM problems. The strategy of Means-Ends Analysis is quite simple, that if we cannot solve a problem in one step, why not solve the problem in multiple steps and each step solve a small proportion of the problem. The first problem for this strategy is how to guaranty that each step is forward to the final goal, not backward to the initial state, which may trap my agent into an infinity loop. There is a solution to solve the problem that we can describe the difference between initial and final states as a quantitative value, and my agent will only consider steps that reduce the number. The detailed calculations of difference between Figure 1C and Figure 1D are illustrated in Table 1.

Table 1

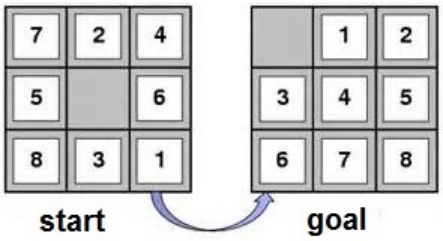
Difference	Initial state	Score	Step 1	Score	Step 2	Score	Goal	Score
Object positions	1 above 2	1	1 above 2	1	1 above 2	1	1 left 2	0
	1 above 3	1	1 above 3	1	1 above 3	1	1 left 3	0
	3 inside 2	1	3 inside 2	1	2 inside 3	0	2 inside 3	0
Objects operations	1 unchanged	1	1 rotated	0	1 rotated	0	1 rotated	0
	2 unchanged	0	2 unchanged	0	2 unchanged	0	2 unchanged	0
	3 unchanged	1	3 unchanged	1	3 expanded	0	3 expanded	0
		5		4		2		0

The object positions and operation information for Figure 1D is obtained through Semantic network as shown in Figure 2. Between Figure 1C, the initial state, and Figure 1D, the goal, there are 3 position differences and two operation differences with the total difference score 5. My agent will perform random operations on objects 1, 2 and 3 in Figure 1C, and among them only two operations (rotation operation on object 1 and expanded operation on object 3) can reduce the difference score. The rotation operation on object 1 reduces difference score from 5 to 4, while the expanded operation on object 3 reduces difference value to 2, and that is because during expanded operation, the related positions between object 2 and object 3 changed. Further operations on objects will not reduce the difference score. Next, my agent will randomly move objects positions, and finally figure out moving object 1 to the left side will reach the goal state.

Means-Ends Analysis is not that simple:

You must realize that I am using an over simplified system to describe the idea of Means-Ends Analysis, and the real situations are much more complicated. One of the difficulties is how to describe the difference between states into a quantitative number, and any step that forwards goal reduce the number while backwards increase the number. That usually involves many mathematicians' diligent work. There is another example, the 8-puzzle problem that ask you to move the number blocks that transform the start state to goal state as shown in Figure 3. There are many solutions, and one of them is called Hamming priority function. That method uses Hamming number to describe the states difference. The Hamming number is the number of blocks at wrong position plus the moves, and only moves that can reduce Hamming number will reach the goal state. Obviously, the idea of using of Hamming number to describe states difference is not that straightforward.

Figure 3



1 2 3 4 5 6 7 8

1 1 1 1 1 1 1 1

Hamming = 8 + 0

Another difficulty of Means-Ends Analysis is that this strategy is hard to do “tactical retreat”, that acknowledging a small defeat in the short term to gain a long-term advantage, since we describe the states difference as a quantitative number and only consider moves that reduce the number. If not designed properly, my agent will act like the following dog (photo is obtained from internet).