Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence

Learning about Representational Modality: Design and Programming Projects for Knowledge-Based AI

Ashok K. Goel, Maithilee Kunda, David Joyner, and Swaroop Vattam

School of Interactive Computing, Georgia Institute of Technology, 85 Fifth Street NW, Atlanta, GA 30308
goel@cc.gatech.edu, mkunda@gatech.edu, david.joyner@gatech.edu, svattam@gatech.edu

Abstract

Many AI courses include design and programming projects that provide students with opportunities for experiential learning. Design and programming projects in courses on knowledge-based AI typically explore topics in knowledge, memory, reasoning, and learning. Traditional AI curricula, however, seldom highlight issues of modality of representations, often focusing solely on propositional representations. In this paper, we report on an investigation into learning about representational modality through a series of projects based around geometric analogy problems similar to the Raven's Progressive Matrices test of intelligence. We conducted this experiment over three years, from Fall 2010 through Fall 2012, in a class on knowledge-based AI. We used the methodology of action research in which the teacher is also the researcher. We discovered that students found these projects motivating, engaging, and challenging, in several cases investing significant time and posting their work online. From our perspective, the projects accomplished the goal of learning about representational modality in addition to knowledge representation and reasoning.

Introduction

AI courses typically include design and programming projects in addition to lectures, discussions, and homework assignments. These design and programming projects serve to enhance learning in at least three ways: (1) they enable active, situated, and experiential learning, (2) they help make abstract AI concepts real, concrete, and tangible, and (3) they provide students with opportunities to explore and experiment with AI concepts and methods on their own. Moreover, student engagement with these projects often is an important contributor to retention in AI as well as in computer science as a whole.

The goals of design and programming projects in courses on knowledge-based AI (KBAI) typically include explorations into knowledge, memory, reasoning and/or learning. Projects on the role of knowledge in AI often focus on the content, representation, organization, acquisition, and use of knowledge. Few such AI projects pertain to representational modality, e.g., propositional vs. iconic representations. This is likely because most AI projects assume that representations are propositional to both (1) situate projects in existing AI systems and literature and (2) make the projects easily accessible to students with a typical programming background and familiarity with propositional data structures.

In this paper, we describe an investigation into learning about representational modality in addition to the more common learning goals in KBAI classes. We used action research to conduct this experiment in a KBAI class over three years, from Fall 2010 through Fall 2012. In action research, the practitioner is also a researcher interested in understanding and improving the practice; improvement occurs through a process of action and critical reflection (Bryndon-Miller et al., 2003; Schon 1984; Yin 2009). When action research is applied to learning science, the teacher is also the researcher, designing interventions in the class, collecting data, and critically reflecting on the results, leading to redesigned or new interventions. In our experiment, the intervention took the form of a series of four open-ended and extended design and programming projects. We collected data through student performance on the projects, class and individual feedback from students, and an anonymous exit survey of students.

A Course on Knowledge-Based AI

CS 4635 and CS 7637 are undergraduate and graduate sections of a 3-credit semester-long course in knowledge-based AI that is offered each fall term at Georgia Tech. The two sections meet together and do the same work. The course typically is taught by the first author (Goel). The

Goel, A. K., Kunda, M., Joyner, D. A., & Vattam, S. (2013). Learning about representational modality: Design and programming projects for knowledge-based AI. Fourth AAAI Symposium on Educational Advances in Artificial Intelligence, AAAI Conference, Bellevue, WA. p. 1586-1591.

class projects in 2010 through 2012 were based on the Ph.D. research of the second author (Kunda); the other two authors (Joyner and Vattam) have taken the course at different times and also served as TAs for the course in 2012.

A typical CS 4635/7367 class may have 30-35 undergrad students, 15-20 M.S. students, and 2-5 Ph.D. students. The university's course catalog describes the contents of CS 4635/7637 as "Structured knowledge representation; knowledge-based methods of reasoning and learning; problem-solving, modeling and design." The course is taught from the perspective of cognitive systems, taking the goals of AI as both building intelligent systems and understanding human intelligence (Langley, 2012).

Raven's Test of Intelligence

The goals of the design and programming projects in the KBAI class are similar to other AI courses of this kind: experiential learning about AI concepts and methods of knowledge, memory, reasoning, and learning. Over the years, the course instructor has often based class projects on research projects in his group, the Design & Intelligence Laboratory (http://dilab.gatech.edu). For the last three years, the design projects have been based on the research of the second author on the Raven's Progressive Matrices test, and they also align with the "psychometric AI" approach (Bringsjord & Schimanski, 2003).

The Raven's Progressive Matrices test is a standardized psychometric test of intelligence that contains geometric-analogy-like problems in which a matrix of figures is presented with one missing entry, and the correct missing entry must be selected from among a set of answer choices (Raven et al., 2003; see Figures 2-4 for sample problems). Despite the fact that Raven's problems are presented in only one format (in contrast to IQ tests containing numerous subtests across different domains), the Raven's test provides an exceptionally strong measure of overall IQ, and in fact is the best such measure of any single-format, single-domain cognitive test (Snow et al., 1984).

Despite the breadth of its use, the specific nature of the information processing used on the Raven's test is not fully understood. It is thought that strategies vary along many dimensions, including representational modality (e.g. Lynn et al., 2004), for instance certain problems might be "visual" problems, solved using visuospatial transformations, while other problems are thought to be "verbal" problems, solved using amodal strategies like logical rule induction. Interestingly, recent findings indicate that individuals with autism seem to solve the entire test visually, for both "visual" and "verbal" problems (Soulières et al., 2009), which aligns with evidence for a general visual bias in certain individuals with autism (Kunda & Goel, 2011).

Hunt (1974) made an early attempt to describe this dissociation in modality by sketching two algorithms, a "Gestalt" algorithm and an "Analytic" algorithm, that corresponded to the "visual" and "verbal" strategies, respectively, observed in human test-takers. Since Hunt's paper, several models of the Raven's test have been developed that align with each of these approaches.

A production system using hand-coded propositional representations of problem inputs remains perhaps the most widely-cited account of information processing on the Raven's test (Carpenter et al., 1990). More recent systems have used logic, propositional pattern matching, and structure mapping (see Kunda et al., 2013, for a detailed review). All of these systems utilize propositional representations for the core part of reasoning.

Two models developed by our research laboratory that operate directly on scanned images from the test have been successful on several versions of the Raven's test (Kunda et al., 2010, 2012, 2013; McGreggor et al., 2011, 2012). One of these models, the ASTI model, uses a combination of affine transformations and set operations on pixel values to emulate the types of operations observed in studies of human mental imagery (Kosslyn et al., 2006). The model first induces a visual transformation to explain the changes that occur across images in the problem matrix, then predicts a candidate image for the empty space in the matrix, and finally compares this candidate image to each of the choices using a ratio model of similarity (Tversky, 1977). The ASTI model was tested on three versions of the Raven's test and achieves scores at or near ceiling on two tests (Kunda, 2013). The ASTI model was developed primarily to illustrate the feasibility of solving Raven's problems using a purely visual modality of representations.

Design and Programming Projects

Given that the Raven's intelligence test encompasses hard issues pertaining to human intelligence, computational psychometrics, analogical reasoning, learning, knowledge representation, and representation modality, the first author hypothesized that the Raven's test could be a good task domain in which to conduct design and programming projects for the KBAI class. We thus decomposed the ASTI model the second author had developed for her Ph.D. work into four open-ended and extended class projects. We first introduced the projects in Fall 2010 and have been improving them each year based our observations of student performance, student feedback, and our assessment of student learning. In Fall 2012, the students had three weeks per project to complete the four project assignments below.

Project 1: Table 1 illustrates Project 1, which used problems from the Miller analogy test, which were used in one of the more well-known early AI programs, Evans' ANALOGY program (Evans, 1964). Note that in addition to the input images (Figure 1), we gave students a sample propositional representation of the input images (Table 2). Note also that Project 1 included three problems given to

Table 1: Text of assignment given for KBAI Project 1 (condensed for space).

Goal: The goal of the project is to learn about the close relationship between learning and problem solving. On one hand, results of learning often help us solve problems. On the other, we often learn what we need to address some problem. The task here is to address A:B::C:x? analogy problems from Miller's test of intelligence.

Deliverables: You will be given three visual reasoning problems. Your goal is to write a propositional representation for each of these problems and a program that can solve these problems based on that propositional representation. As such, you will deliver:

1. Propositional representations (a text file) for all three problems.
2. A computer program that loads those propositional representations and solves the problems, written in Java, Python, or C#.
3. A report that explains and justifies the design of the architecture and algorithms of your program and the experiments you conducted. Make sure to comment on efficiency: how much extra time will be involved if problem complexity expands?

We will supply you with the visual representations of the three problems, but these are only for your benefit - your program will not read them. Instead, you should use these visual representations to construct your own propositional representations which will act as the input. The output, in turn, is your program's answers to each of the three problems.

Fourth Problem: After the due date, a fourth problem will be supplied. You will have one week to return a propositional representation of this fourth problem. The goal here is for your original project (without modifying the source code) to be sufficiently general that it can approach new problems that are structured according to your propositional representation schema.

Running the Program: When executed, your program should follow the following guidelines:

- The program must automatically begin to solve the problems when run, without requiring input parameters.
- Your program can run with either a GUI or text interface, but it must display some visual sign of ongoing progress.
- Your program must display the answers to all solved problems in clearly-readable format at the conclusion of the run.

Grading: Grades will be assigned with a 20% weight for correctly solving each of the four problems, with an additional 20% for the written report, including an evaluation of the efficiency and generalizability of the algorithm.

Notes: The 20% each for problems 1 through 4 is not an all-or-nothing check if the algorithm chose the answer we say is right. If the algorithm has sound reasoning to choose a "wrong" answer (such as the rotation vs. mirroring in Problem 1), it will receive just as much credit. Similarly, if the algorithm arrives at the right answer using unsound reasoning, full credit won't be given.

- Your representation can describe the internal structure of each frame of the problem in any way you want; for example, it is acceptable to make statements like "rectangle outside triangle" or "triangle inside triangle". You may not, however, describe the relationship between the frames. For Problem 2, for example, you may not include in your representation, "rectangle moves outside triangle". Determining the relationships between frames is the job of the algorithm.

students at the start, and a fourth problem given once students had turned in their program. Students could represent the fourth test problem using the input representation their program accepted but could not change the program code. Since students did not know the fourth problem in advance, they needed to make their programs general enough to address new problems similar to the original three.

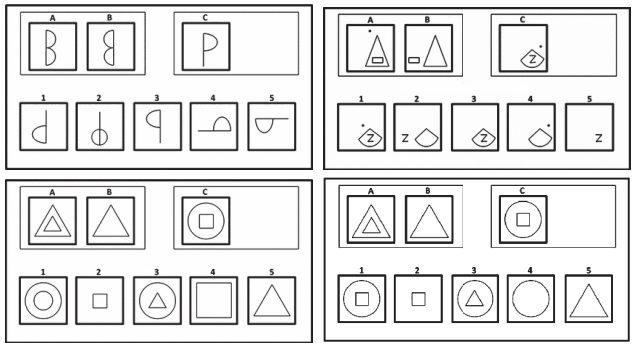

Figure 1: Miller analogy problems for Project 1. The fourth problem was given after students completed their programs.

In addition to their programs, students were asked to turn in design reports that explained the architecture and algorithms of their program as well as any theoretical analysis (e.g., time and space complexity) or experimental analysis (e.g., ablation experiments) they may have done.

Table 2: Part sample propositional representation for Project 1.

Attribute is a (slot, filler) pair. Syntax: «X» represents an object.
^foo bar sets the object's variable foo to the value bar.
^foo bar baz sets foo to a list containing the values bar and baz.

```
// Evidence A
(«e1»
^index A
^num-figures 2
^figures «e1-f1» «e1-f2»)
(«e1-f1»
^index 1
^evidence «e1»
^num-attributes 3
^attributes «e1-f1-a1»
«e1-f1-a2» «e1-f1-a3»)
(«e1-f1-a1»
^index 1
^figure «e1-f1»
^slot shape
^filler triangle)
(«e1-f1-a2»
^index 2
^figure «e1-f1»
^slot location
^filler center)
(«e1-f1-a3»
^index 3
^figure «e1-f1»
^slot size
^filler big)
```

Project 2: While Project 1 solved 2x1 matrix problems, Project 2 addressed 2x2 and 3x3 problems (Figure 2). Further, while Project 1 gave three test problems initially and one later, Project 2 gave six problems initially and two problems later (one for extra credit).

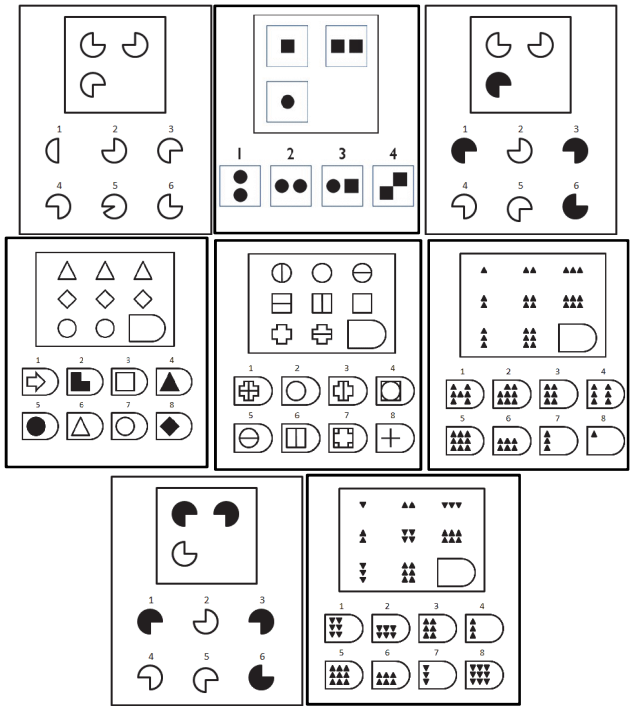

Figure 2: Raven's-like problems for Project 2. The last two problems were given after students completed their programs.

Project 3: Unlike Projects 1 and 2, Project 3 asked students to work directly with the input images. We gave six problems initially (the same as the first six in Project 2) and another two new problems later on (Figure 3). Here is a summary of what the assignment said about inputs:

"The key difference in this project is that rather than loading the textual propositional representations, your program will load the visual representations themselves. However, this is not a class on image processing, and thus you have flexibility in your visual representations. For example, you are welcome to manually split the problem into multiple frames for the program to read (i.e., your representation for Prob. 1 could have 9 individual images).

Generally, there are many ways that the visual input could be done. Do not worry too much on how to actually load the image into the program; any method that you can conceive of that allows the program to operate on the individual pixels is acceptable. You can also consider each pixel either black or white, rather than greyscale."

Note that Project 3 does not specify the modality of the internal representation of the program. The program could work directly with input images at the pixel level (iconic representations) or extract propositions over which to reason, as in earlier projects. To introduce students to reasoning with iconic representations, the second author gave a guest lecture about her research on the ASTI model.

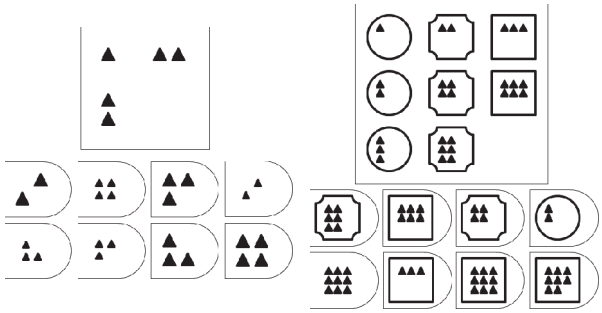

Figure 3: Raven's-like problems for Project 3, given after students completed their programs based on Project 2 problems.

Project 4: Project 4 combined propositional and visual reasoning strategies. As before, there was an initial set of problems to guide students' program development, plus two new problems to test the generality of their approaches. Here is a summary of the assignment:

"The first goal of the project is to complete a program for addressing ALL TEN problems from Projects 2 and 3. However, this time, your program should implement BOTH methods of reasoning, (a) propositional reasoning and (b) visual reasoning. In order to keep the comparison between the two strategies honest, you want to use the same image input for both. This means that:

(a) In Project 3, if you used the visual reasoning method, then you will need to create another method to extract propositions from the image inputs and work with those propositional representations.

(b) If, on the other hand, your program in Project 3 already extracts propositions from the input images, then you will need to implement another method that embodies visual reasoning without extracting propositions.

The second goal of the project is to compare the two methods on the same set of input problems with respect to both accuracy and performance of the two methods. This comparative analysis should discuss which method is better suited for which class of problems and why."

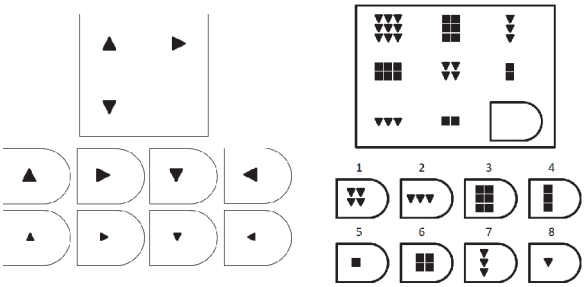

Figure 4: Raven's-like problems for Project 4, given after students completed their programs based on previous problems.

Discussion

Grading: Each of the four projects counted for 10% of the semester grade, thus together accounting for a large proportion of the grade. We graded each project on a 0-100 scale that was, as mentioned in Project 1, distributed equally among the various problems as well as the quality of written reports. Grades were assigned with a focus on the quality of the problem solving enacted by students' programs, rather than whether their programs achieved some objective "correct" answer. The weight assigned to the design report gradually increased from the first (20%) to the fourth projects (35%). We reduced 40% of the grade for projects that were up to one day late, 70% for two days, 90% for three days, and 100% for more than three days late. We gave "extra credit" to students who did extra work such as making up and addressing new problems. In addition to numerical scores, we tried to give written feedback on the students' programs as well as design reports.

Student performance: A little to our surprise, we discovered that most students found the projects based on the Raven's test motivating, engaging and challenging. Many students made great effort at not only completing the projects, but also doing extra work. Some students reported that they had spent dozens of hours on each project.

In Fall 2012, the median scores in the four projects in the undergrad section of the KBAI course were 99, 96, 95 and 85 out of 100, respectively; the scores in the grad section were 100, 102, 104 and 93, respectively. Some scores were higher than 100 because extra credit was offered for completing an additional, more challenging problem, as well as for particularly comprehensive, well-written design reports. We believe the slightly lower scores on the fourth project are due to the increased difficulty in obtaining this extra credit, as well as routine end-of-semester time limitations. Overall, we think that these scores are quite high and speak to the high quality of work by the students. From a teacher's perspective, we believe that the projects accomplished our learning goals for the students.

Examples of student work: Two students posted their work on their personal websites (without prompting from us). We share links to these not only to provide examples of the students' work, but also to illustrate that these students felt a sense of ownership and accomplishment strong enough to want to post their projects online. The first of the two students was in the undergrad section in 2010 and is now a graduate student at CMU; the second student was in the grad section in 2011 and is doing his Ph.D. in computer vision at Georgia Tech. Note that their solutions pertain to slightly different problems from the 2012 ones described here because we have revised and refined the projects and problems each year.

- http://main.sauvikdas.com/research/projects/22-vis-analogy.html?start=4
- http://www.cc.gatech.edu/~vbettada/files/Ravens-VisualAndPropReasoning.pdf

Student feedback: Georgia Tech collects optional student evaluations of all courses and instructors at the end of each term, anonymizes responses, and shares them with course instructors. In Table 3, we include both positive and negative comments pertaining to the projects from both sections of the course in Fall 2011 and Fall 2012.

In addition to the formal written feedback in Table 3, we sought informal oral feedback on the projects in the class, during office hours, and in the hallways. On the positive side, several students told us that they found the projects challenging in both their difficulty and the questions they raised about human and artificial intelligence. Some students thought that it was "thrilling" to see their program address problems from an intelligence test that many of the students themselves had taken at one point in their career. Many students said that their computer programs made them introspect on their own intelligence. One TA reports that several students came to discuss implementation plans for multiple hours each during the semester to brainstorm the most robust and comprehensive approaches, often going beyond the scope of the graded assignments. On multiple occasions, students implemented complex features that the TA specifically said were not required for full credit, solely out of personal engagement.

On the negative side, some students found the projects very time consuming. Some students expressed that the projects were monotonous. Perhaps the most critical comments concerned the relationship of the projects with the rest of the course: several students told us that while the projects were interesting and useful, they did not relate well with the course material as a whole. On reflection, we find this criticism as legitimate: much of the rest of the course covers traditional areas in knowledge-based AI that emphasize only propositional representations. Two ways of addressing this mismatch might be to emphasize the representational, reasoning, and learning aspects of the projects in addition to the modality aspects, as well as to explicitly describe and situate the learning goals for each individual project within the larger course context.

Conclusion

Our investigations into KBAI design and programming projects based on the Raven's Progressive Matrices test of intelligence lead us to three preliminary conclusions:

(1) The KBAI projects based on the Raven's test enable experiential learning about basic concepts of knowledge, representation, memory, reasoning and learning, including frames and analogy. Future iterations will address student concerns about making connections between the projects and the rest of the course more explicit.

Table 3: Written feedback from students about the projects from Fall 2011 and Fall 2012.

Positive Feedback	Negative Feedback
• The projects were very interesting. • It opened my mind to a lot of new ways of thinking about AI and the functions AI systems could provide to the world. • It was interesting material, and I really enjoyed the projects. • The projects. As much work as they took, they made me feel like a real computer scientist. • I enjoyed the visual vs prepositional approaches • Projects were research-level and interesting; lots of topics and new research areas covered • I liked that we could choose the language for the projects. • Good mix of homework assignments, class discussion and projects. • The structure of the project helps learning and understanding of the topics. • They were right in the nice, meaty area of challenging-but-solvable. He gave us very little guidance, which forced us to truly consider the problems and create novel solutions. • Assignment present an interesting micro-research problem, solving which is a helpful experience in conducting research in KBAI. • A very good thing is that we had no predefined way to solve the problem - instead, creativeness was welcomed. • The Projects... They were difficult and helped learn a lot!	• Projects very hard. • The projects take upwards of 12 hours/week to complete so just be forewarned. They are on par with the machine learning course's projects. • Maybe one or even two projects could be cut down, that would reduce the workload from the course. Also in place of the projects, I would rather have home-works as they stimulated our reasoning process and focused on the immediate concepts being taught in the class. Minimum two projects must be included in the coursework as they make the course interesting but four is a bit too much and they make the course pretty time-intensive one. • The projects need to provide more variety. i felt like I was completing the same project over and over. That got boring very fast. • Additionally, the projects did not prepare us for the mid-term in any way. • I think the last couple assignments were little bit easier, It could have been made more exciting. • The projects did not have much to do with the material covered.

(2) These projects also enable exploration and experimentation with representation modality, a topic often neglected in educational curricula for knowledge-based AI.

(3) Most students found the projects motivating, engaging, challenging and intriguing. Many students were thrilled to find they could write a computer program that can address problems from an intelligence test. Some AI students are fascinated by the projects through being stimulated to introspect about their own intelligence.

References

Bringsjord, S., & Schimanski, B. (2003). What is artificial intelligence? Psychometric AI as an answer. In Procs. *IJCAI*, pp. 887–893.

Bryndon-Miller, M., Greenwood, D., Maguire, P. (2003) Why Action Research? *Action Research* 1(1): 9-28.

Carpenter, P.A., Just, M., & Shell, P. (1990). What one intelligence test measures: a theoretical account of the processing in the RPM Test. *Psychological Review*, 97 (3), 404-431.

Evans, T. G. (1964) A program for the solution of a class of geometric-analogy intelligence-test questions. Cambridge AFRL.

Hunt, E. (1974). Quote the raven? Nevermore! In *Knowledge and cognition* (pp. 129–158). Hillsdale, NJ: Erlbaum.

Kosslyn, S., Thompson, W., & Ganis, G. (2006). *The case for mental imagery*. New York, NY: Oxford University Press.

Kunda, M. (2013). Visual Problem Solving in Autism, Psychometrics, and AI: The Case of the Raven's Progressive Matrices Test. Ph.D. Dissertation, School of Interactive Computing, Georgia Institute of Technology.

Kunda, M., & Goel, A. (2011). Thinking in Pictures as a cognitive account of autism. *J.Autism & Developmental Disorders*. 41 (9): 1157-1177.

Kunda, M., McGreggor, K., & Goel, A. (2010). Taking a look (literally!) at the Raven's intelligence test: Two visual solution strategies. In *Procs. 32nd Annual Conference of the Cognitive Science Society*, pp. 1691-1696.

Kunda, M., McGreggor, K., & Goel, A. (2012). Reasoning on the Raven's Advanced Progressive Matrices test with iconic visual representations. In *Procs. 34th Annual Conf. Cognitive Science Society*, pp. 1828-1833.

Kunda, M., McGreggor, K., & Goel, A. (2013). A computational model for solving problems from the Raven's Progressive Matrices intelligence test using iconic visual representations. *Journal of Cognitive Systems Research*, 22-23, 47-66.

Langley, P. (2012) The Cognitive Systems Paradigm. *Advances in Cognitive Systems*, 1:3-13.

Lynn, R., Allik, J., & Irwing, P. (2004). Sex differences on three factors identified in Raven's SPM. *Intelligence*, *32* (4), 411-424.

McGreggor, K., & Goel, A. (2012) Fractal Analogies for General Intelligence. In *Procs. Fifth International Conference on Artificial General Intelligence*, LNAI 7716: 177-188.

McGreggor, K., Kunda, M., and Goel, A. 2011. Fractal Analogies: Preliminary Results from the Raven's Test of Intelligence. In *Procs. Second International Conf. Computational Creativity,* Mexico City, Mexico.

Raven, J., Raven, J. C., & Court, J.H. (2003). *Manual for RPM and Vocabulary Scales*. San Antonio, TX: Harcourt Assessment.

Schon, D. (1984) *The Reflective Practioner: How Professionals Think in Action*. Basic Books.

Snow, R. E., Kyllonen, P. C., & Marshalek, B. (1984). The topography of ability and learning correlations. *Advances in the Psychology of Human Intelligence: Volume 2*, 47-103.

Soulières, I., Dawson, M., Samson, F., et al. (2009). Enhanced visual processing contributes to matrix reasoning in autism. *Human Brain Mapping*, *30* (12), 4082-4107.

Tversky, A. (1977). Features of similarity. *Psych. Review*, *84*(4), 327-352.

Yin, R. (2009). *Case Study Research, Design and Methods*, Sage Publications, Thousand Oaks, CA.