Proceedings of the Ninth Annual Conference on Advances in Cognitive Systems Submitted 9/2021; published 11/2021

Automatic Item Generation of Figural Analogy Problems: A Review and Outlook

Yuan Yang YUAN.YANG@VANDERBILT.EDU
Deepayan Sanyal DEEPAYAN.SANYAL@VANDERBILT.EDU
Joel Michelson JOEL.P.MICHELSON@VANDERBILT.EDU
James Ainooson JAMES.AINOOSON@VANDERBILT.EDU
Maithilee Kunda MKUNDA @VANDERBILT.EDU
Electrical Engineering and Computer Science, Vanderbilt University, Nashville, TN 37235 USA

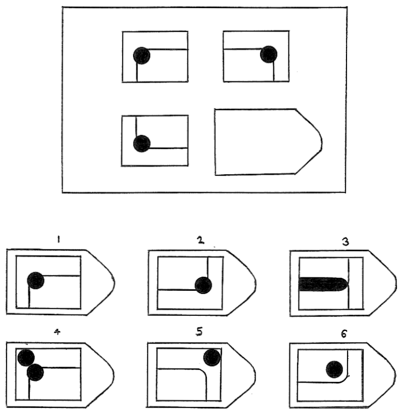
Abstract

Figural analogy problems have long been a widely used format in human intelligence tests. In the past four decades, more and more research has investigated automatic item generation for figural analogy problems, i.e., algorithmic approaches for systematically and automatically creating such problems. In cognitive science and psychometrics, this research can deepen our understandings of human analogical ability and psychometric properties of figural analogies. With the recent development of data-driven AI models for reasoning about figural analogies, the territory of automatic item generation of figural analogies has further expanded. This expansion brings new challenges as well as opportunities, which demand reflection on previous item generation research and planning future studies. This paper reviews the important works of automatic item generation of figural analogies for both human intelligence tests and data-driven AI models. From an interdisciplinary perspective, the principles and technical details of these works are analyzed and compared, and desiderata for future research are suggested.

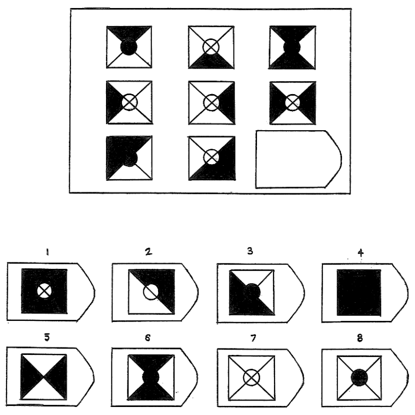
1. Introduction

Figural analogy (FA) tasks (sometimes called geometric analogy, matrix reasoning, etc.) have long occupied a very central position among human intelligence tests (Snow et al., 1984), and have also had a long history of research in artificial intelligence (AI).An FA is usually represented as a 2D array of figures. Figure 1 shows 4 common formats of figural analogy problems (FAPs). Figure 1a and 1b exemplify the Raven's Progressive Matrices (RPM), which is widely used as the single intelligence test whose score is most representative of the general intellectual ability. Figure 1c shows another FA task — proportional geometric analogies — that were attempted by the first computational solver of FAPs (Evans, 1964). Figure 1d shows another common FA task in intelligence tests — figural series.

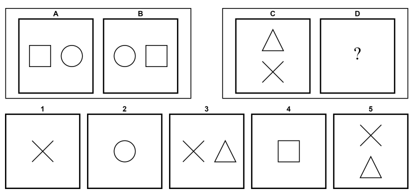
Many computational models for solving analogy problems have been constructed in different disciplines, such as psychometrics, neuroscience, cognitive science, and artificial intelligence. Although these models are different in their purposes and theoretical foundations depending on what disciplines they belong to, they do share a great demand for high-quality analogy problems that



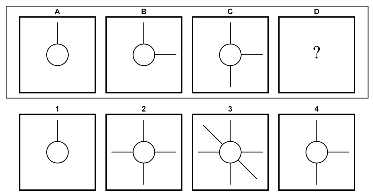
(a) 2×2 Progressive Matrix (Kunda et al., 2013)



(b) 3×3 Progressive Matrix (Kunda et al., 2013)



(c) 2+2 Geometric Analogy (Lovett et al., 2009)



(d) 4×1 Figural Series (Sekh et al., 2020)

Figure 1: Examples of figural analogy problems.

should be as diverse and representative as possible in their particular analogy domain. In particular, since the data-driven approaches in AI joined the game, this demand has soared up exponentially, and instantly dwarfed the reserve of analogy items and the productivity of human item writers of analogy problems.

The purpose of this paper is to seek possible solutions to this supply-demand gap in a different line of research — automatic item generation (AIG). AIG refers to the approach of using computer algorithms to automatically create testing items. It has been adopted and studied for decades in areas such as psychology, psychometrics, cognitive science, and education. It generates a wide range of testing items from domain-general tests, such as human IQ tests (Hornke & Habon, 1986), to domain-specific tests, such as medical license tests (Gierl et al., 2012). AIG was initially introduced to address the increased demand for testing items in the following test settings:

- Large-scale ability testing, for example, repeated tests in academic settings and longitudinal experiments, where many parallel forms are needed due to the retest effect.
- Adaptive testing, in which the next items are determined by the responses to previous items, which is a more efficient and reliable testing form, but also requires larger item banks.
- Computer-based and internet-based testing, which makes standardized tests more accessible to the public and brings the exposure control issue to a new level.

In addition, recent work on data-driven FA approaches in AI creates another type of demand — datasets to fit and evaluate AI models. More importantly, as the usage of testing items in AI may be vastly different from that in human testing, this presents a new research question about how AIG should evolve to properly serve all the research purposes in these different usages — constructing AI models, understanding human cognition, and evaluating both AI and human performance.

In this paper, we will first review the methods of automatically generating FAPs for human intelligence tests. Then, we switch to the works of automatically generating FAPs for fitting and evaluating data-driven AI models. However, our review is by no means exhaustive, and we select the most representative ones to analyze. At last, we conclude with reflections on the reviewed works and suggest possible future directions for AIG of FAPs.

2. Automatically Generating Figural Analogy Problems for Human Intelligence Tests

Human intelligence tests consist of items carefully handcrafted by psychologists and psychometricians. However, item writing is more of an art than a science due to its idiosyncrasy and inefficiency. Handcrafted items have to go through iterations of evaluation and calibrating for good psychometric properties before being included in the final item bank. The attrition rate could be up to 50% (Embretson, 2004). A variety of efforts in AIG have been made to free item writers from the onerousness of item writing. In this section, we discuss the important works of automatically generating FAPs for intelligence tests, each of which is described in a subsection, whose title is the name of the work followed by a keyword of its most outstanding characteristic. The technical details of the works are summarized in Table 1.

2.1 Rule-Based Item Construction — Human-Based AIG

Hornke & Habon (1986) conducted one of the earliest studies, if not the earliest, on AIG. Although they did not literally automatically generate items, what they achieved was already very close to today's AIG. They created a standard procedure for item generation, hired university students to manually execute this procedure, and created 648 3×3 FAPs. Each step in this procedure had finite clearly defined options so that human idiosyncrasy was kept at the minimum level. To the best of our knowledge, although the diversity and complexity of these items were not comparable to expert designs, no one had ever "automatically" created so many items.

Hornke & Habon considered the item writing task as the reverse of the solving process, which could be decomposed into three types of cognitive operations. The three types addressed three common independent dimensions of solving or creating an FAP item. To generate items, Hornke & Habon thus designed a procedure that sequentially made choices on these three dimensions by selecting from finite sets of options:

- Variation rules of geometric elements: eight options are provided (see the first 8 matrices in Figure 2 for examples): identity, addition, subtraction, intersection, exclusive union (or symmetric difference), progression, variation of open/closed gestalts (i.e. distribution of three hollow/solid shapes).
- Analogical directions: a variation rule proceeds in row or column direction.

Table 1: The technical details of generator programs in the reviewed works.

Model	Format	Analogical Direction[a]	Geometric Element						Rules		Per. Org.[d]	Answer Set[e]
			Type[b]	Size	Color	Angle	Number	Position	Type[c]	Number		
Rule-Based	3×3	R, C, O	-	-	-	-	1-2	-	I, S, P, D	1-2	S, I, E	-
Cognitive	3×3	R, C, O	-	-	-	-	-	-	I, S, P, D	1-2	O, F, D	8
GeomGen	3×3	R, C	S, C,T, D2, H, R	-	-	-	-	3×3 grid	I, S, P, N	-	C, N	5+1
Sandia	3×3	R, C, M, S, O	O, R, T, D1, T1, T2	5 sizes	5 colors	5 angles	1-6	center	S, P	1-3	center, overlay	8
CSP	3×3	R, C	-	-	-	-	-	-	S, P, D	1-7	-	8
IMak	2×2	O	C, L, D0, T1	1 size	1 color	8 angles	3	-	P	1-4	E	8+2
PGM-shape	3×3	R, C	C, T, S, P, H	10 sizes	10 colors	-	0-9	3×3 grid	S, P, D	1-4	3×3 grid	8
PGM-line	3×3	R, C	L, D1, C	1 size	10 colors	1 angle	1-5	center	S, P, D	1-4	center, overlay	8
RAVEN	3×3	R	T, S, P, H, C	6 sizes	10 colors	8 angles	1-9	fixed in configurations	I, S, P, A, D	4	7 configurations	8

[a] R=row, C=column, M=main-diagonal, S=secondary-diagonal, O=R+C (see (Matzen et al., 2010)).

[b] S=square, C=circle, R=rectangles, T=triangles, H=hexagons, O=oval, L=line, D0=dot D1=diamond, T1=trapezoid, T2="T", D2=deltoid, P=pentagon..

[c] I=identify, S=set/logical operations, P=progression, D=Distribution of 2 or 3 values, N=neighborhood, A = arithmetic (see (Carpenter et al., 1990; Arendasy, 2002)).

[d] S=separation, I=integration, E=embedding, C=classical, N=normal, O=overlay, F=fusion, D=distortion (see (Hornke & Habon, 1986; Embretson, 1998; Primi, 2001)).

[e] +1="none of the above", +2="none of the above" + "I don't know".

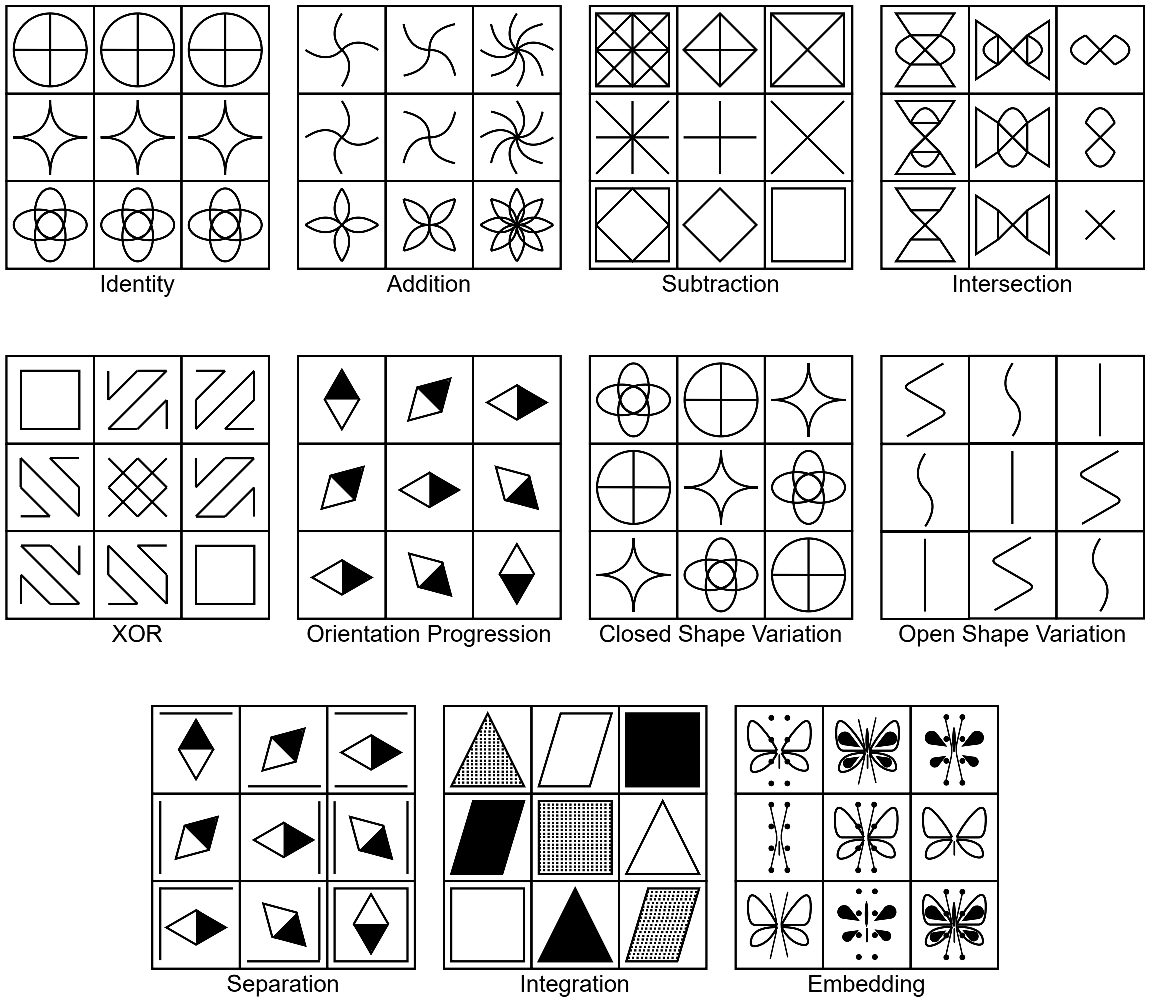


Figure 2: Example items created by following Hornke & Habon's AIG procedure.

- Perceptual organizations: this dimension addresses how multiple variation rules are combined into a single item. Three options are provided (see the last 3 matrices in Figure 2 for examples): separation, integration, and embedding. Separation means that separate geometric elements are used for different variation rule; integration means that different attributes of a single geometric element are used for different variation rules; and embedding means that different parts of a single geometric element are used for different variation rules.

In their experiment, student item writers were given a set of geometric elements (e.g. differently sized squares and triangles) and instructed to create items by jointly sampling the 3 dimensions and geometric elements from the given set. The students were told to create each items by combining at most two variation rules. Therefore, the resulting item bank contained only 1-rule and 2-rule items. Human experiments on this item bank showed that the cognitive operations corresponding to these

3 dimensions explained roughly 40% of the item difficulty. As for the unexplained 60%, other early studies (Mulholland et al., 1980) indicated that the numbers of elements and rules were also major sources of difficulty.

2.2 Cognitive Design System Approach — Cognitive Modeling

Embretson (1995, 1998, 2004) introduced the Cognitive Design System Approach. Based on the information processing theory of Carpenter et al. (1990), an item bank of FAPs was generated as a demonstration. The Cognitive Design System Approach starts with the cognitive modeling of the solving process of an existing ability test. Embretson pointed out that the cognitive model in (Carpenter et al., 1990), which many following AIG works were based upon, did not include perceptual encoding or decision processes in the solving process. Hence Embretson incorporated three extra binary perceptual stimulus features — object overlay, object fusion, and object distortion — in the Cognitive Design System Approach, which represent three different types of mental decomposition of the complete gestalt into its basic parts. Essentially, object overlay and fusion are similar to separation and embedding in Figure 2, while object distortion refers to perceptually altering the shape of corresponding elements (e.g. bending, twisting, stretching, etc.). A software, ITEMGEN, was developed based on this approach, but the software is not easily accessible now.

Once the cognitive models are determined, the stimulus features are accordingly determined. The Cognitive Design System Approach then employs psychometric models to estimate item properties (such as item difficulty and item discrimination), formulated as parameterized functions of the stimulus features. The function parameters are initially set by fitting the psychometric models to human data on the existing tests. Thereafter, the psychometric properties of newly generated items (by manipulating the stimulus features) could be predicted by these functions. The prediction and empirical analysis of the newly generated items are compared to further calibrate the parameters until the functions are sufficiently predictive. The approach could be integrated into an adaptive testing system to replace a fixed item bank and generate items of expected properties in real-time.

2.3 MatrixDeveloper — 4×4 Matrices

MatrixDeveloper (Hofer, 2004) is an unpublished software that generates FAPs. It has been used in several studies of psychometric properties of computer-generated FAPs (Freund et al., 2008; Freund & Holling, 2011a,b,c). According to the limited description in these studies, the MatrixDeveloper is similar to the Cognitive Design System Approach in terms of variation rules (i.e. the five rules summarized in (Carpenter et al., 1990)) and perceptual organizations (i.e. overlap, fusion, and distortion). The difference is that it generates 4×4 matrix items. Therefore, it could accommodate more variation rules compared to 3×3 or 2×2 matrices so that the differential effects of types of variation rules could be better studied.

2.4 GeomGen — Perceptual Organization

The early cognitive modelings of solving hand-crafted FAPs tended to characterize the items by the numbers of elements and rules and types of rules, for example, (Mulholland et al., 1980; Bethell-Fox et al., 1984; Carpenter et al., 1990). However, unlike solving the existing items, creating new items requires us to consider at least one more factor — perceptual organization (Primi, 2001). It tells how geometric attributes, geometric elements and rules are perceptually integrated to render the item image. For example, the third dimension in the procedure of (Hornke & Habon, 1986) is a specific way to deal with perceptual organization. More generally, perceptual organization involves the Gestalt grouping/mapping of elements using Gestalt principles such as proximity, similarity, and continuity. This factor is less clearly defined and less systematically investigated (and whether it is a unidimensional factor is even questionable). But, to create new items, one has to adopt some formalized ways to manipulate perceptual organization.

(Arendasy, 2002; Arendasy & Sommer, 2005) proposed a generator program — GeomGen — that adopted a binary perceptual organization, which was reused and extended in many following works. The perceptual organization in GeomGen was classified into classical view and normal view. In classical views, the appearance of geometric elements changes while numbers and positions of them remain constant. In normal views, numbers and positions of elements change while the appearance remain constant. The appearance changes, such as in size and aspect ratio or distortions, should preserve the Gestalt grouping/mapping of visual similarity. Therefore, in both classical and normal views, the type (i.e. geometric shape) of elements are set to be constant in GeomGen.

The taxonomy of perceptual organization in GeomGen is only a specific way to define perceptual organization but by no means the unique way. For example, Primi (2001) proposed another important taxonomy — harmonic and nonharmonic, which, together with GeomGen taxonomy, forms a better description of perceptual organization, which is adopted in many following works.

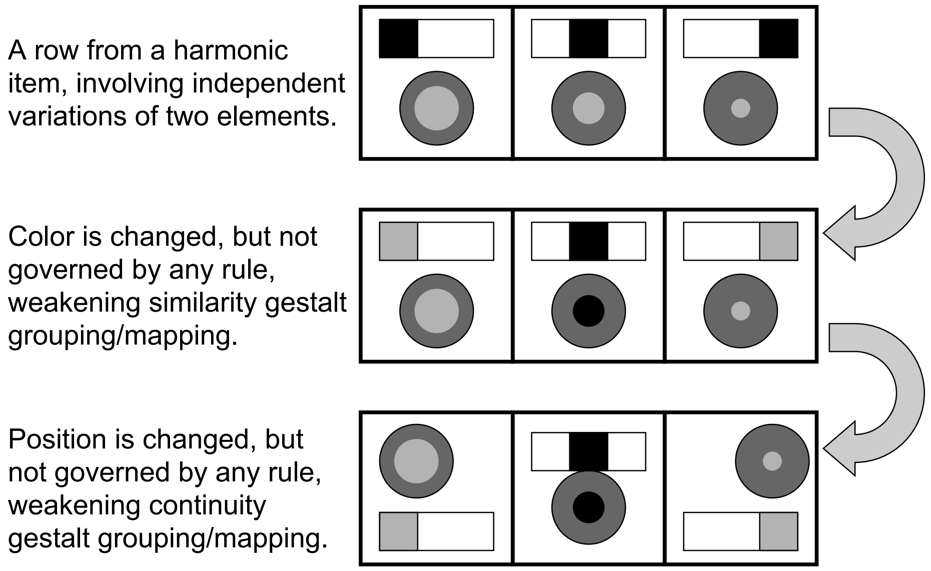

A row from a harmonic item, involving independent variations of two elements.
Color is changed, but not governed by any rule, weakening similarity gestalt grouping/mapping.
Position is changed, but not governed by any rule, weakening continuity gestalt grouping/mapping.

Figure 3: An example of deriving nonharmonic items from harmonic items.

Primi (2001) classified perceptual organizations into harmonic and nonharmonic ones— "visually harmonic items display perceptual and conceptual combinations that represent congruent relationships between elements, whereas nonharmonic items tend to portray competitive or conflicting combinations between visual and conceptual aspects that must be dealt with in reaching a solution." Primi (2001) mentioned that, in the practice of AIG, the nonharmonic items could be derived from the harmonic ones by manipulating the attributes of elements to cause misleading Gestalt groupings, as shown in Figure 3. The correct Gestalt grouping/mapping (i.e. element correspondences) are obvious in harmonic items, whereas nonharmonic items requires extra cognitive effort to resolve the conflict between competing gestalt groupings and mappings.

2.5 Sandia Matrix Generation Software — High-Fidelity SPM Generator

Based on the Raven's Standard Progressive Matrices (SPM), Matzen et al. (2010) presented the Sandia Matrix Generator that could "recreate" the 3×3 SPM items with high fidelity.

Matzen et al. (2010) identified two basic types of 3×3 items in SPM— the element transformation and the logic problems. An element transformation refers to a progressive variation of a certain attribute of the element. There could be multiple variations in different directions, for example, a color variation in the row direction and a size variation in the column direction. The attributes considered for transformation problems are shape, shading, orientation, size, and number, each of which takes values from an ordered categorical domain. The logic problems involve operations such as addition/subtraction, conjunction (AND), disjunction (OR), or exclusive disjunction (XOR) of elements along rows. The elements in logic problems are overlaid with one another. Each generated item can be either a transformation one or a logic one, but not both.

The Sandia Matrix Generator generates answer choices in ways that are common among the original SPM problems. An incorrect answer choice could be (a) an entry in the matrix, (b) a random transformation of an entry in the matrix, (c) a random transformation of the correct answer, (d) a random transformation of an incorrect answer, (e) a combination of features sampled from the matrix, and (e) a combination of novel features that did not appear in the matrix.

The item difficulty was studied through an item bank of 840 generated items. Human experiment data showed that the generated items and the original SPM had generally very similar item difficulty. In particular, the data further showed that the item difficulty was mostly affected by the number of rules, analogical directions, and problem types.

2.6 CSP Generator — First-Order Logic Representation

Wang & Su (2015) made an effort to represent 3×3 RPM items in a more formal manner — the first-order logic — and turned the AIG problem into a constraint satisfaction problem (CSP) by formulating the "validity" of RPM items into a set of first-order logic propositions.

In particular, an variation rule is represented as an instantiation of Equation (1) and (2),

$$\exists \alpha \ \forall i \in \{1,2,3\} \ \exists o_{i1}, o_{i2}, o_{i3} \ P(\alpha, o_{i1}, o_{i2}, o_{i3}) \tag{1}$$

$$\begin{aligned} P(\alpha, o_{i1}, o_{i2}, o_{i3}) = {} & Unary(\tau(o_{i1},\alpha), \tau(o_{i2},\alpha), \tau(o_{i3},\alpha)) \wedge \\ & Binary(\tau(o_{i1},\alpha), \tau(o_{i2},\alpha), \tau(o_{i3},\alpha)) \wedge \\ & Ternary(\tau(o_{i1},\alpha), \tau(o_{i2},\alpha), \tau(o_{i3},\alpha)) \end{aligned} \tag{2}$$

where α is an attribute, i is the row index, o_{ij} is a geometric elements in the figure of Row i and Column j, $\tau(\alpha, o_{ij})$ is the value of α of o_{ij}, and P is a predicate that describes the variation pattern of attribute α in each row. In Equation (2), the predicate P further equals a conjunction of three predicates — $Unary$, $Binary$, and $Ternary$ — representing three categories of relations commonly used in 3×3 RPM problems, as illustrated in Figure 4.

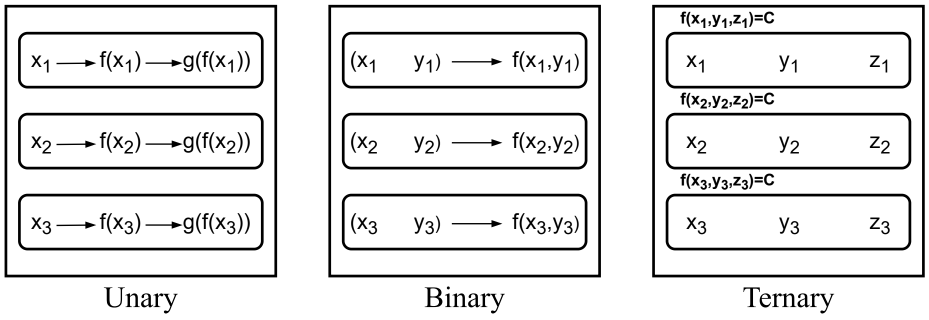


Figure 4: Three categories of relations commonly used in 3×3 RPM problems (Wang & Su, 2015).

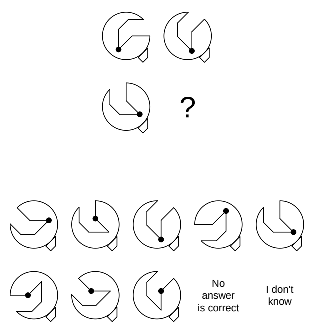
An interesting observation of Figure 4 is that, mathematically, the unary relation is a proper subset of the binary relation, which is a proper subset of the ternary relation. That is, the ternary relation is thus mathematically sufficient to generate all the items. However, interpreting the same variation as unary, binary and ternary relations probably requires different working memory abilities and goal management abilities. Therefore, these three categories are probably cognitively “disjoint”, and need to be separately included in a generator program to achieve a better control over psychometric properties.

Equation (1) and (2) represent only the variation pattern of a single attribute α. There could be multiple variation patterns of different attributes in a matrix. It is possible that some attributes are not assigned instantiations of Equation (1) and (2), and they could be given either constant values or random values across figures. Random values may cause distracting effects in generated items, which is similar to the nonharmonic perceptual organizations in (Primi, 2001).

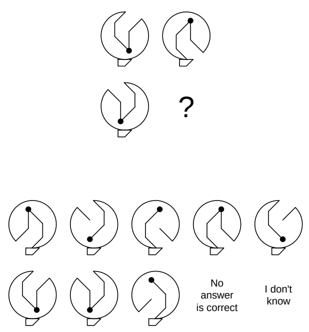
Human data on the generated (non-distracting) items showed that the overall difficulty and rule-wise difficulty (number of rules) were similar to the items in Advanced Raven’s Progressive Matrices (APM). However, as the author pointed out, their generator could not synthesize all the items in APM for some underlying transformations were hard to implement. When the items were created with distracting attributes, the generated items became much more difficult for human subjects.

2.7 IMak Package — Open Source

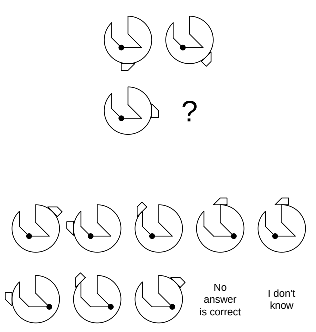
Blum & Holling (2018) released IMak — an open source R package — to study how different types of variation rules could affect item difficulty. The generator was designed to manipulate the types of rules while keeping other factors constant, and the generated items look quite different from what one would expect in ordinary FAPs, such as RPM items. For example, Figure 5 shows example items that we created through the package, each of which exemplifies a basic rule type. With the current release (version 2.0.1), the geometric elements are limited to the main shape (the broken circle plus the polyline in it), the trapezium that is tangent to the main shape, and the dot at one of the corners of the polyline. Furthermore, the size and shape of each element are fixed for all generated items, but the position, orientation and existence would vary in finite domains according to 5 basic rules.



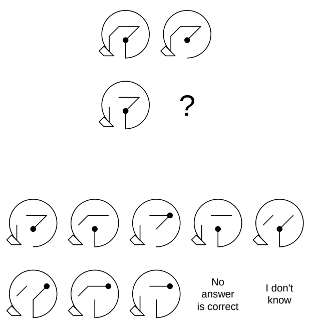
(a) Main shape rotation



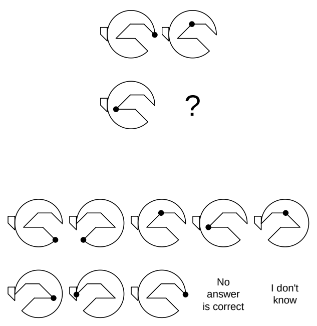
(b) Main shape reflection



(c) Trapezium rotation



(d) Line segment subtraction



(e) Dot movement

Figure 5: Example items generated by the IMak package. Each item exemplifies a single basic rule. The correct answer for each item is arranged as the first option for demonstration purposes.

IMak has four other distinctive features. Firstly, IMak filled the vacancy of 2×2 format of AIG. Being affected by the famous work of (Carpenter et al., 1990) on RPM, the vast majority of AIG works would only consider the 3×3 format. However, the original RPM test also contains 2×2 items, and the proportional geometric analogies (see the 2+2 format in Figure 1c) could also be put into the 2×2 format. Secondly, the answer set contains two more meta-options “no correct answer” and “I don’t know”, which encourage subjects to solve the items more constructively rather than eliminating responses. Thirdly, the variation of one element could depend on the variation of another element. For example, the dot’s moves depend on the variation of the main shape, since the dot only moves along the polyline in the main shape. This kind of variation is very rare in item writing of FAPs, and it represents an extra level of complexity of FAP items.

Last but not least, IMak used a rule-dependent strategy to generate incorrect answer choices. For 1-rule items, 4 distinct values of the attribute of the rule are sampled, including the correct value; since all other attributes remain constant in the matrix, another random attribute is chosen and sampled for 2 values. The resulting 8 (4×2) combinations make the 8 options in the answer set. For 2-rule items, 3 values are sampled for each of the 2 attributes of the 2 rules, resulting in 9 combinations, and one of them is discarded. For 3-rule items, 2×2×2 combinations are sampled. For four-rule items, 2×2×2×2 combinations were sampled, and half of them are discarded.

In a human experiment, 23 generated items were administered to 307 participants from Germany, Indonesia, and Argentina. Reliability, validity and unidimensionality were initially verified by the experiment results. Particularly, item difficulty could be **partly** predicted from the number and type of rules based on psychometric models, but far from enough for psychometric applications.

3. Automatically Generating FAPs for Fitting Data-Driven AI Models

3.1 Procedurally Generated Matrices

Based on the five rules in (Carpenter et al., 1990), Barrett et al. (2018) continued the first-order logic approach of Wang & Su (2015) and created a large-scale (1.2M) dataset of FAPs — Procedurally Generated Matrices (PGM). Since the generator program is not made public, our discussion is solely based on the description in (Barrett et al., 2018) and our observation of the dataset.

In PGM, an instantiation of Equation (1) and (2) in the first-order logic approach was denoted by a triplet $[r, o, a]$ of relation, object and attribute. Object here means the geometric element in previous sections of AIG for human intelligence tests. But since most reviewed works in this section use object instead of element, we follow this convention in this section. These three factors are not independent, and we visualized their dependencies in Figure 6, which contains 29 paths from the left to the right, corresponding to 29 $[r, o, a]$ triplets. This number equals the number of triplets mentioned in (Barrett et al., 2018) (but it did not provide a list of the 29 triplets).

As shown in Figure 6, the objects in PGM are classified into two disjoint subsets — shape and line. In the shape subset, closed shapes are arranged in 3×3 grid (fixed positions in this case) in each figure. In the line subset, line drawings, spanning the whole figure, are always centered in each figure. There exist items generated by superimposing a shape item on a line item, but these two items are completely independent in terms of variations. Thus, in Table 1, we split PGM into two rows to describe it more clearly.

The generation procedure of a PGM item could be described by 5 steps: (a) sample 1 to 4 triplets from the 29 triplets described in Figure 6. (b) determine the analogical direction for each triplet: row or column; (c) sample attribute values for each triplet from their domains. (d) determine the attribute values for unspecified attributes (either constant or random). and (e) render all attribute values into a pixel image of the matrix.

3.2 Relational and Analogical Visual rEasoNing

Due to the limited expressive power of $[r, o, a]$ in PGM, the spatial configuration, as an important dimension of perception organization, is highly restricted — 3×3 grid for the shape subset, all-centered for the line subset, and superimposing a shape item on a line item for the intersection of these two subsets. Zhang et al. (2019a) attempted to encode the spatial configuration and generated the Relational and Analogical Visual rEasoNing (RAVEN) dataset, which included 7 hardcoded spatial configurations, as shown in Figure 7. The source code of RAVEN's generator is available online, inspection of which drives our discussion.

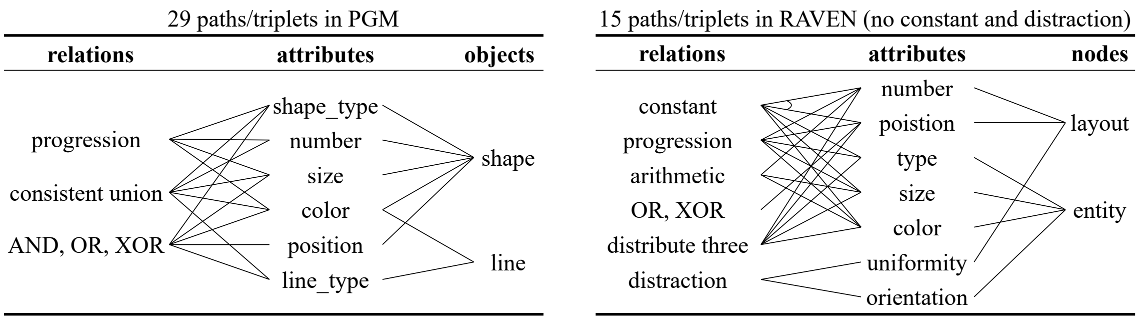


Figure 6: Left: The dependencies among relations, objects, and attributes used to generate the Procedurally Generated Matrices (PGM) dataset (Barrett et al., 2018). Each path corresponds to a $[r, o, a]$ triplet representing a possible type of variation pattern in the matrices. As one can check, there are 29 paths, i.e. $[r, o, a]$ triplets, in the graph. Note that in Barrett et al. (2018) did not differentiate between "shape_type" and "line_type" and referred to both of them as "type". In their implementation, these two attributes are treated as two distinct ones. Right: The dependencies among relations, nodes, and attributes were used to generate the RAVEN dataset. Note that we listed "distraction" as a rule in this graph to indicate that uniformity and orientation are distracting attributes. The two paths from constant through number and position to layout are treated as a single rule in RAVEN. Therefore, there are 15 paths, but 14 rules, in the graph.

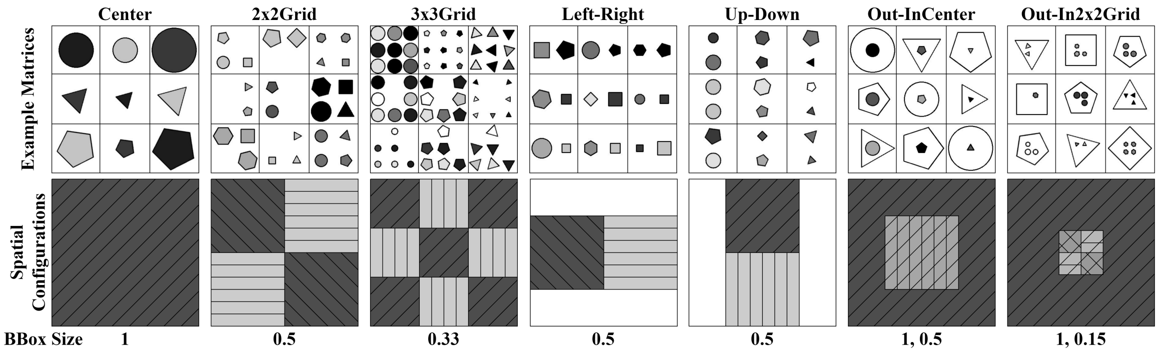


Figure 7: 7 hardcoded spatial configurations — center, 2x2Grid, 3x3Grid, Left-Right, Up-Down, Out-InCenter, and Out-In2x2Grid — are used to arrange objects in each figure in the RAVEN dataset. Each configuration is represented by the bounding boxes that objects could occupy. The position and size of each bounding box are hardcoded in the generator program. An example matrix for each configuration is given in the first row. Note that not every bounding box has to be occupied, but every object has to be in one of the bounding boxes.

The 7 configurations are derived from a more general symbolic representation framework of images — Attributed Stochastic Image Grammar (A-SIG). In A-SIG, a potential image is described by a tree structure, where the conceptual granularity becomes finer and finer toward the leaf nodes. To generate RAVEN, the tree structure is predefined as a general A-SIG tree as shown in Figure

8, which consists of 5 conceptual levels — scene, structure, component, layout, and entity — and uses a stochastic tree-traversal process to generate items. However, the 7 configurations in RAVEN were hardcoded in the language of A-SIG, rather than generated through this stochastic traversing process, which could have made RAVEN more diverse in spatial configuration, and, meanwhile, more challenging.

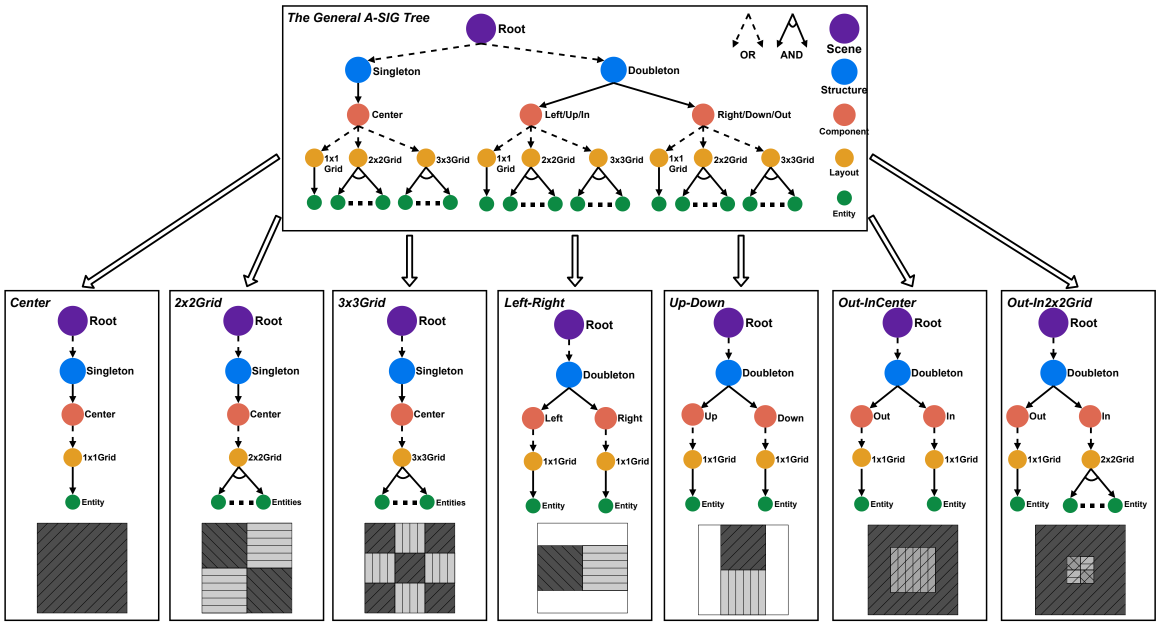


Figure 8: The general A-SIG tree and 7 specific A-SIG trees were used in the RAVEN dataset (image adapted from (Zhang et al., 2019a) by adding technical details from the source code of the generator program). The root node denotes the scene that the images of interest would describe. The structure nodes are the containers of different spatial structures. A structure is composed of components that could be overlaid with each other. Each component has its own layout and, more importantly, variation rules, which are independent of other components. The layout node, as its name indicated, contains the attributes specifying the number and positions of geometric objects. Entities represent geometric objects whose attributes are invariant to number and position.

To compare with the PGM dataset, we represent PGM items also in A-SIG, as shown in Figure 9. The line configuration of PGM is basically the same as the center configuration of RAVEN except that the entity types (shape) are different. The shape configuration of PGM is almost the same as the 3x3Grid configuration of RAVEN except that bounding box sizes are slightly different. The shape-over-line configuration of PGM is also conceptually similar to the double-component configurations of RAVEN. The general difference between PGM and RAVEN lies in the layout and entity nodes. As shown in Figure 9, the PGM dataset is not able to separate the concepts of entity and entity layout due to the limited expressive power of triplets $[r, o, a]$. That is, the object o takes the roles of both layout and entity nodes, but could not play the roles effectively and simultaneously.

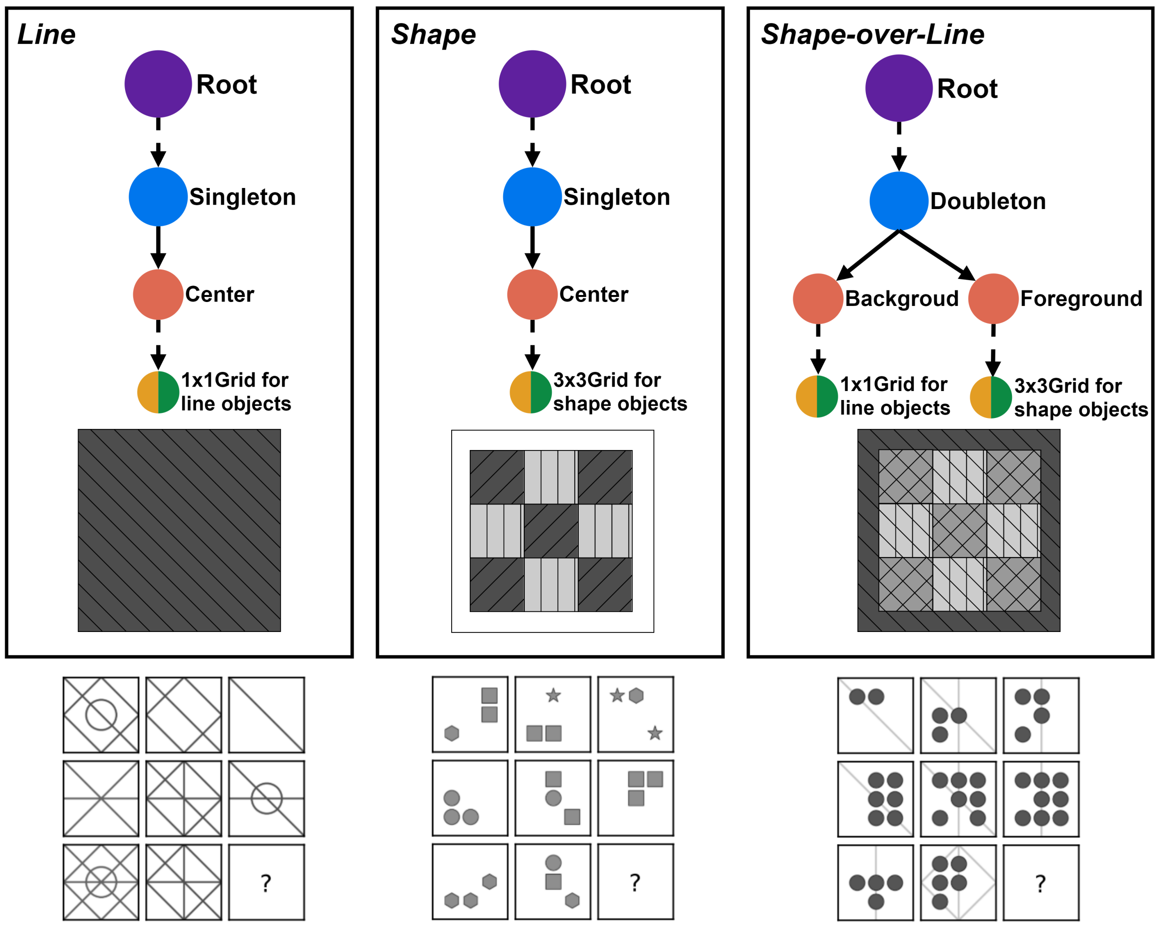


Figure 9: The spatial configurations of PGM could also be represented in A-SIG to compare with RAVEN. Based on observations, there are 3 spatial configurations in PGM — line, shape, and shape-over-line. The example matrix is given for each configuration at the bottom.

Unlike PGM, RAVEN inherited all the five rules from (Carpenter et al., 1990). Moreover, the "addition-and-subtraction" rule type in RAVEN contains not only figure addition and subtraction, such as the set operations "OR and XOR", but also arithmetic addition and subtraction, which were not in (Carpenter et al., 1990). Since these two operations are conceptually different, we refer to the arithmetic addition and subtraction as "arithmetic", and the figure addition and subtraction as "OR and XOR". In addition, the "distribution-of-three-values and distribution-of-two-values" in (Carpenter et al., 1990) were merged into a single rule by considering the latter as a special case of the former with a null value for one of the three values. Therefore, RAVEN has a slightly different rule set compared to PGM. Similarly, we could represent the variation rules of RAVEN also as triplets — $[r, n, a]$ where n represents nodes (layout or entity) in A-SIG trees, and r and a are relations and attributes, being the same as PGM. Then Figure 6 shows the dependencies among r, n and a.

PGM and RAVEN are similar in some respect, and meanwhile, different in others. In particular, they share two similarities. First, their choices of attributes, attribute domains, and rule types are similar. For example, they both forbid number-rule and position-rule from co-occurring in an item because these two attributes would probably conflict with each other. Second, although RAVEN has more spatial configurations, these configurations are not structurally different from PGM (as can be seen from the comparison of their A-SIG trees).

Meanwhile, PGM and RAVEN are different in two respects. First, they are different in the number of rules in an item. In PGM, 1 to 4 triplets were sampled from the 29 triplets. In contrast, in a RAVEN item, every attribute is governed by a rule except the two distracting attributes (uniformity and orientation). Thus, there are 4 rules (for number/position, type, size, and color, respectively) in each RAVEN item. Second, the rules in RAVEN are all row-wise while the rules in PGM are either row-wise or column-wise.

3.3 Context-Blind Issue

The answer sets in RAVEN were generated in the same way as the first-order logic approach. Each incorrect answer was obtained by modifying a single attribute of the correct answer. The method kept the maximum level of distracting and confusing effect of incorrect answer choices, because one has to identify all the variation rules to solve the problem. Meanwhile, ignoring any rule would lead to multiple choices. However, this design has a major drawback for failing the context-blind test. In a RAVEN item, the incomplete matrix could be considered as the context that provides information for solving the problem. Failing the context-blind test means that it is possible for human subjects or computational models to select the correct answer while turning blind to the context.

Hu et al. (2021) and Benny et al. (2021) pointed out that RAVEN failed the context-blind test. They provided evidence that computational models could achieve high accuracies (from 70%+ to 90%+) by training only on the answer sets of RAVEN. In addition, the context-blind accuracies were even better than the accuracies that could be achieved by training the same data-driven model on complete items. This implies that data-driven AI models are capable of capturing the statistical regularities in the answer sets. The reason for this context-blind issue obviously lies in the answer set generation, in which every incorrect answer choice is a variant by modifying a single attribute of the correct answer choice. Therefore, the correct answer is the answer choice that possesses every common feature among all the choices (or, equivalently, the one most similar to every other choice).

Both Hu et al. (2021) and Benny et al. (2021) proposed their own solutions to this issue — the Impartial-RAVEN and RAVEN-FAIR datasets. These two datasets have the same context matrices as the original RAVEN and regenerated the answer sets in different ways. Interestingly, the similarity and difference among these three versions of answer sets could be clearly illustrated by putting them in graphs (simple graph, not digraph or multigraph). If we represent each answer choice as a vertex and each modification of a single attribute as an edge, then the answer sets of the three versions could be depicted by the graphs in Figure 10. The answer set of the original RAVEN is created by modifying an attribute of the correct answer. Thus, its graph is a star centered at the correct answer (the solid vertex). And what the aforementioned computational models captured was the unique center of this star graph.

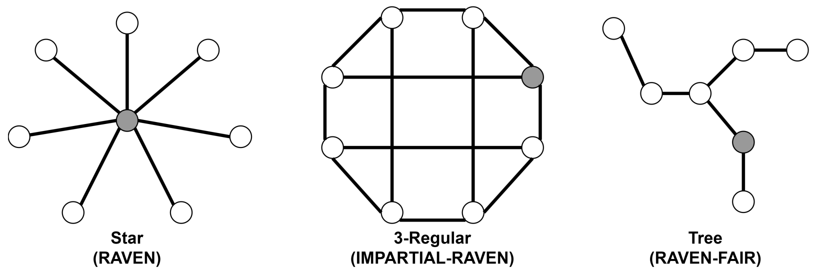


Figure 10: The answer sets of three versions of the RAVEN dataset depicted in graphs. Each vertex is an answer choice and two adjacent vertices differ by one attribute.

Hu et al. (2021) proposed the Impartial-RAVEN whose each answer set could be represented by a 3-regular graph in Figure 10. Benny et al. (2021) proposed another less regulated way to generate answer sets. Starting from an initial answer set consisting of only the correct answer, an answer choice is randomly selected from the current answer set, and then an attribute of the selected answer choice is randomly altered; this process repeats, resulting in a random tree-like structure as shown on the right in Figure 10.

These two enhanced versions of RAVEN were tested by context-blindly training the benchmark model in (Zhang et al., 2019a) and the CoPINet model in (Zhang et al., 2019b). The performance decreased to below 20%. Ideally, a human subject (or a computational model) who context-blindly works on the RAVEN items should perform at most as good as a random guess, i.e. $1/8 = 12.5\%$, which implies that the answer set per se does not provide any useful information for solving the item. However, in the practice of item writing, to maintain a certain level of distracting and confusing effect of incorrect answer choices, the majority of incorrect answer choices must share some similarities with the correct answer. Otherwise, it would be relatively easy for subjects to find the correct answer, because incorrect answers would be perceptually distinct from the context. Therefore, a reasonable context-blind performance would be slightly higher than the random guess, and determined by the researcher's judgment.

An interesting observation on the difference between these two enhancements could be made by comparing their graphs in 10. If we consider a single trial (in a probabilistic sense) where we context-blindly give a subject (or an AI model) an item from Impartial-RAVEN and an item from RAVEN-FAIR, the probability that this subject solves the Impartial-RAVEN item would not be significantly different from that of solving the RAVEN-FAIR item. However, if we repeat this trial (with different items) again and again, the performance on RAVEN-FAIR would probably gradually exceed the performance on Impartial-RAVEN, of course, assuming that the subject is intelligent enough and able to figure out the graph structures, and thus makes an educated guess by selecting the "center" (or the max-degree vertex) of trees in a probabilistic sense. In this case, we would say that the RAVEN-FAIR is context-blind valid at the item level, but not at the dataset level.

4. Discussion and Conclusion

In this paper, we reviewed automatic item generation (AIG) of figural analogy problems (FAPs). We classified the works into two groups by their purposes — whether it is for human ability tests or for fitting and evaluating AI models. The works in the first group aim at not only generating items but also good psychometric properties. The works in the second group could be seen as the continuation of the works in the first group, but the psychometric aspects were usually not stressed.

For example, 18 PGM items were administered to human participants. Participants without prior experience of FAPs would often fail almost all the items, whereas participants with prior experience could score up to 80%. Therefore, the discriminative power is too large and the ability range that could be effectively tested is too narrow. This also implies that the datasets used to evaluate data-driven AI models are not necessarily suitable to evaluate human cognitive ability.

More importantly, this gives rise to another interesting question — how do we assess the analogical ability of AI models trained and evaluated on the large datasets in the second group? On one hand, some data-driven AI models indeed performed well on AIG items that posed great challenges to human subjects; on the other hand, training on the large-scale datasets specially prepared the AI models for a highly restricted subset of the problem domain, but human subjects, who were not trained at all, or just trained on several representative exemplars from this subset, could perform well and even generalize well in the problem domain.

This is not the first time that this question is asked, for example, (Detterman, 2011). Efforts have been made to address this issue by many researchers. (Bringsjord & Schimanski, 2003; Bringsjord, 2011) tried to address this issue by incorporating the implementation and evaluation of AI models into a general concept of psychometric AI. By reviewing roughly 30 computational models for solving intelligence tests, Hernández-Orallo et al. (2016) proposed that (a), instead of collecting items, we should collect item generators, and (b) the generated items should be administered to machine and human (and other animals) alike (universal psychometrics). All these propositions are constructive and, meanwhile, suggest much higher requirements for AIG studies.

In the domain of FAPs, which is loosely (even vaguely) defined, a practical roadmap for future AIG studies could better start from studying the problem domain per se instead of continuing to construct ad hoc generator programs. Huge uncharted territories lie in the complexity factors such as the types of elements and rules and perceptual organization. Current AIG item banks and datasets are far below the level of the flexibility and diversity that human item writers can achieve. For example, the spatial configurations in PGM and RAVEN are hardcoded and basically stack independent items together; inter-element variation, in which the variation of one element depends on the variation of another element, is also very rare, as are perceptually and conceptually ambiguous analogies.

At last, our review is by no means comprehensive and exhaustive; and a combined interdisciplinary effort is definitely a desideratum for future research.

References

Arendasy, M. (2002). Geomgen—ein itemgenerator für matrizentestaufgaben. *Wien: Eigenverlag*.

Arendasy, M., & Sommer, M. (2005). The effect of different types of perceptual manipulations on the dimensionality of automatically generated figural matrices. *Intelligence*, *33*, 307–324.

Barrett, D., Hill, F., Santoro, A., Morcos, A., & Lillicrap, T. (2018). Measuring abstract reasoning in neural networks. *International Conference on Machine Learning* (pp. 511–520). PMLR.

Benny, Y., Pekar, N., & Wolf, L. (2021). Scale-localized abstract reasoning. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 12557–12565).

Bethell-Fox, C. E., Lohman, D. F., & Snow, R. E. (1984). Adaptive reasoning: Componential and eye movement analysis of geometric analogy performance. *Intelligence*, *8*, 205–238.

Blum, D., & Holling, H. (2018). Automatic generation of figural analogies with the imak package. *Frontiers in psychology*, *9*, 1286.

Bringsjord, S. (2011). Psychometric artificial intelligence. *Journal of Experimental & Theoretical Artificial Intelligence*, *23*, 271–277.

Bringsjord, S., & Schimanski, B. (2003). What is artificial intelligence? psychometric ai as an answer. *IJCAI* (pp. 887–893). Citeseer.

Carpenter, P. A., Just, M. A., & Shell, P. (1990). What one intelligence test measures: a theoretical account of the processing in the raven progressive matrices test. *Psychological review*, *97*, 404.

Detterman, D. K. (2011). A challenge to watson. *Intelligence*, *39*, 77–78.

Embretson, S. E. (1995). The role of working memory capacity and general control processes in intelligence. *Intelligence*, *20*, 169–189.

Embretson, S. E. (1998). A cognitive design system approach to generating valid tests: Application to abstract reasoning. *Psychological methods*, *3*, 380.

Embretson, S. E. (2004). Measuring human intelligence with artificial intelligence: Adaptive item generation. In *Cognition and intelligence: Identifying the mechanisms of the mind*, 251–267. Cambridge University Press.

Evans, T. G. (1964). *A program for the solution of a class of geometric-analogy intelligence-test questions.*. Technical report, Air Force Cambridge Research Labs.

Freund, P., & Holling, H. (2011a). How to get really smart: Modeling retest and training effects in ability testing using computer-generated figural matrix items. *Intelligence*, *39*, 233–243.

Freund, P., & Holling, H. (2011b). Retest effects in matrix test performance: Differential impact of predictors at different hierarchy levels in an educational setting. *Learning & ind. diff.*, *21*, 597–601.

Freund, P., & Holling, H. (2011c). Who wants to take an intelligence test? personality and achievement motivation in the context of ability testing. *Personality & Ind. Diff.*, *50*, 723–728.

Freund, P. A., Hofer, S., & Holling, H. (2008). Explaining and controlling for the psychometric properties of computer-generated figural matrix items. *Appl. Psychol. Meas.*, *32*, 195–210.

Gierl, M. J., Lai, H., & Turner, S. R. (2012). Using automatic item generation to create multiple-choice test items. *Medical education*, *46*, 757–765.

Hernández-Orallo, J., Martínez-Plumed, F., Schmid, U., Siebers, M., & Dowe, D. L. (2016). Computer models solving intelligence test problems: Progress and implications. *Artificial Intelligence*, *230*, 74–107.

Hofer, S. (2004). Matrixdeveloper [unpublished computer software]. Unpublished software.

Hornke, L. F., & Habon, M. W. (1986). Rule-based item bank construction and evaluation within the linear logistic framework. *Applied psychological measurement*, *10*, 369–380.

Hu, S., Ma, Y., Liu, X., Wei, Y., & Bai, S. (2021). Stratified rule-aware network for abstract visual reasoning. *Proceedings of the AAAI Conference on Artificial Intelligence* (pp. 1567–1574).

Kunda, M., McGreggor, K., & Goel, A. K. (2013). A computational model for solving problems from the raven's progressive matrices intelligence test using iconic visual representations. *Cognitive Systems Research*, *22*, 47–66.

Lovett, A., Tomai, E., Forbus, K., & Usher, J. (2009). Solving geometric analogy problems through two-stage analogical mapping. *Cognitive science*, *33*, 1192–1231.

Matzen, L. E., Benz, Z. O., Dixon, K. R., Posey, J., Kroger, J. K., & Speed, A. E. (2010). Recreating raven's: Software for systematically generating large numbers of raven-like matrix problems with normed properties. *Behavior research methods*, *42*, 525–541.

Mulholland, T. M., Pellegrino, J. W., & Glaser, R. (1980). Components of geometric analogy solution. *Cognitive psychology*, *12*, 252–284.

Primi, R. (2001). Complexity of geometric inductive reasoning tasks: Contribution to the understanding of fluid intelligence. *Intelligence*, *30*, 41–70.

Sekh, A. A., Dogra, D. P., Kar, S., Roy, P. P., & Prasad, D. K. (2020). Can we automate diagrammatic reasoning? *Pattern Recognition*, *106*, 107412.

Snow, R. E., Kyllonen, P. C., Marshalek, B., et al. (1984). The topography of ability and learning correlations. *Advances in the psychology of human intelligence*, *2*, 103.

Wang, K., & Su, Z. (2015). Automatic generation of raven's progressive matrices. *Proceedings of the 24th International Conference on Artificial Intelligence* (pp. 903–909).

Zhang, C., Gao, F., Jia, B., Zhu, Y., & Zhu, S.-C. (2019a). Raven: A dataset for relational and analogical visual reasoning. *CVPR* (pp. 5317–5327).

Zhang, C., Jia, B., Gao, F., Zhu, Y., Lu, H., & Zhu, S.-C. (2019b). Learning perceptual inference by contrasting. *Proceedings of the 33rd International Conference on Neural Information Processing Systems* (pp. 1075–1087).