Journal of Problem Solving

Algorithmic Puzzles: History, Taxonomies, and Applications in Human Problem Solving

Anany Levitin[1]

[1] *Villanova University*

Correspondence:
Correspondence concerning this article should be addressed to Anany Levitin, via email to alevitin@villanova.edu.

Keywords:
Keywords: algorithmic puzzles, problem solving, insight

The paper concerns an important but underappreciated genre of algorithmic puzzles, explaining what these puzzles are, reviewing milestones in their long history, and giving two different ways to classify them. Also covered are major applications of algorithmic puzzles in cognitive science research, with an emphasis on insight problem solving, and the advantages of algorithmic puzzles over some other classes of problems used in insight research. The author proposes adding algorithmic puzzles as a separate category of insight problems, suggests 12 specific puzzles that could be useful for research in insight problem solving, and outlines several experiments dealing with other cognitive aspects of solving algorithmic puzzles.

1. INTRODUCTION AND HISTORICAL HIGHLIGHTS

Algorithmic puzzles are puzzles that require design or analysis of algorithms. In other words, these are puzzles that involve, explicitly or implicitly, clearly defined procedures for solving them. We start with a brief review of the long history of algorithmic puzzles, highlighting its major milestones and their applications. In Sections 2 and 3, respectively, we discuss two ways to classify algorithmic puzzles: by the question they pose and by the generality of their input. Section 4 deals with cognitive science applications of algorithmic puzzles, with an emphasis on insight problem solving. Possible future work is discussed in Section 5; there we list 12 algorithmic puzzles that could be useful for research in insight problem solving and suggest 6 experiments dealing with other issues such as solving puzzles by brute force and working backwards, transfer questions, and a board coloring impact. We end the paper with a summary of its content in the "Conclusion" section.

Three river-crossing puzzles in *Propositiones ad Acuendos Juvenes* (*Problems to Sharpen Youths*),[1] attributed to Alcuin of York (ca. 735–804 CE), one of the leading scholars of the Carolingian Renaissance, have commonly been pointed to as the earliest examples of algorithmic puzzles. The most well known of the three is the Wolf, Goat, and Cabbage problem, whose variations have been found not only in other European countries but also in several African cultures (Ascher, 1990). But Petković (2009, p. 2) mentioned that what is now known as the Josephus Problem appeared in Ambrose of Milan's book ca. 370 CE. A version of this puzzle is quoted below in Section 2.

[1] Singmaster and Hadley (1992) provided an annotated translation of *Propositiones ad Acuendos Juvenes* from Latin to English.

The next important algorithmic puzzle appeared in *Libra Abaci* (*The Book of Calculation*), published in 1202 by Leonardo of Pisa, known later by his nickname Fibonacci:

> A certain man had one pair of rabbits together in a certain enclosed place, and one wishes to know how many are created from the pair in one year when it is the nature of them in a single month to bear another pair, and in the second month those born to bear also. (Sigler, 2002, p. 404)

The solution to this puzzle is given by a remarkable sequence called the Fibonacci numbers by the prominent French mathematician Édouard Lucas (1842–1891): 1, 1, 2, 3, 5, 8, . . . Not only do the Fibonacci numbers appear unexpectedly in the natural world, but they also have many interesting mathematical properties that continue to be discovered more than 800 years since Fibonacci's time (see, for example, articles in the *Fibonacci Quarterly* published since 1963). Also, quite a few recreational problems have been designed based on properties of this remarkable sequence (e.g., Knott, 2017).

The next milestone in the history of algorithmic puzzles had to wait for the great Swiss mathematician Leonhard Euler (1707–1783). In 1735, Euler proved that it was impossible to walk through all the seven bridges of Königsberg, an old Prussian city on the banks of the Pregel River, without crossing the same bridge more than once (Figure 1, see next page).

Using modern terminology, Euler reduced the problem to a question about the existence of a path in a graph that traverses all its edges exactly once. The solution to this puzzle is considered the cornerstone of both graph theory and topology. Among numerous modern applications of graph theory in particular, one of the more important is neural networks,

http://dx.doi.org/10.7771/1932-6246.1194

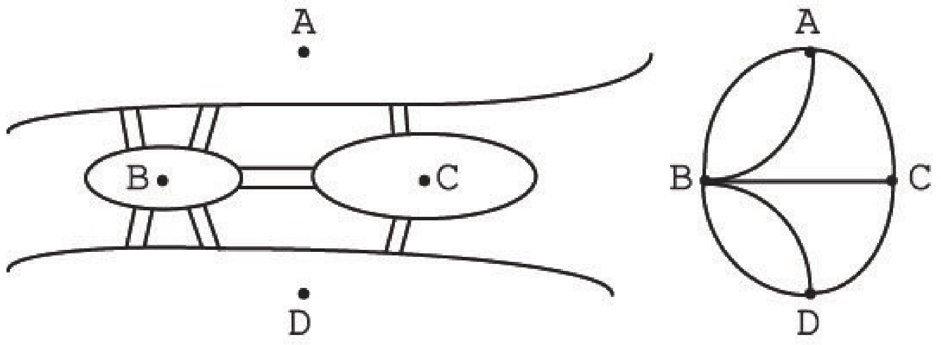

Figure 1.
The seven bridges of the old Königsberg and corresponding graph.

which have advanced studies of brain functions and major neurological diseases (e.g., Bullmore & Sporne, 2009).

A century later, the prominent Irish mathematician and astronomer William Hamilton (1805–1865) invented the Icosian Game to illustrate results of his algebraic discoveries. The object of this one-player game was to find a path visiting all the vertices of a dodecahedron exactly once before returning to the path's starting vertex (Figure 2).

When posed for an arbitrary graph, the existence problem of such a path, called a Hamiltonian cycle, has turned out to be very difficult. (In technical terms it is known to be *NP-complete* [Garey & Johnson, 1979].) The complexity of the Hamiltonian cycle problem is particularly surprising, because the similar question about the existence of a cycle that traverses all the edges of a graph exactly once, called nowadays a Eulerian cycle, has a simple answer given by Euler himself.

In 1883, Éduoard Lucas invented a puzzle that he called the Tower of Hanoi. It consists of three rods and a number of disks of different sizes that can slide onto any rod. Initially all the disks are on the first rod in order of size, the largest on the bottom and the smallest on top (Figure 3, see next page). The objective is to transfer all the disks to the third rod, using the second one as an auxiliary if necessary. Only one disk can be moved at a time, and it is forbidden to place a larger disk on top of a smaller one.

The recursive algorithm solving the puzzle has provided an early example of an algorithmic problem with a straightforward recursive solution and no obvious nonrecursive solutions, although several nonrecursive algorithms were later discovered. The puzzle has also proved to be of great value for different experiments in human problem solving, which we are going to review briefly in Section 4.

The Game of Life, invented by the British American mathematician John Horton Conway and popularized by Martin Gardner in his October 1970 "Mathematical Games" column in *Scientific American*, ought to be considered the most important algorithmic puzzle of the 20th century. This solitaire game starts with a collection of "life" cells marked on an infinite two-dimensional board. After that, a sequence of new configurations called "generations" is obtained by the following rules, which are applied simultaneously to every cell in the current generation. Every cell interacts with its eight neighbors, which are the cells that are adjacent to it horizontally, vertically, or diagonally. To get a new generation, the following transitions occur:

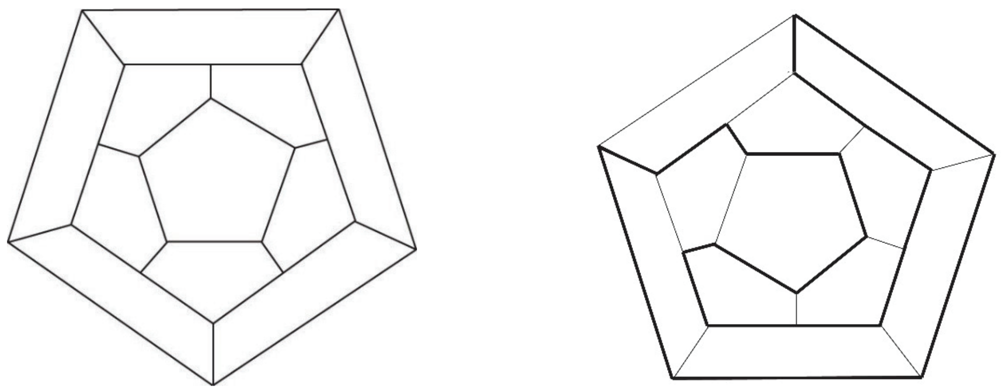

Figure 2.
The board of the Icosian Game and one of its solutions.

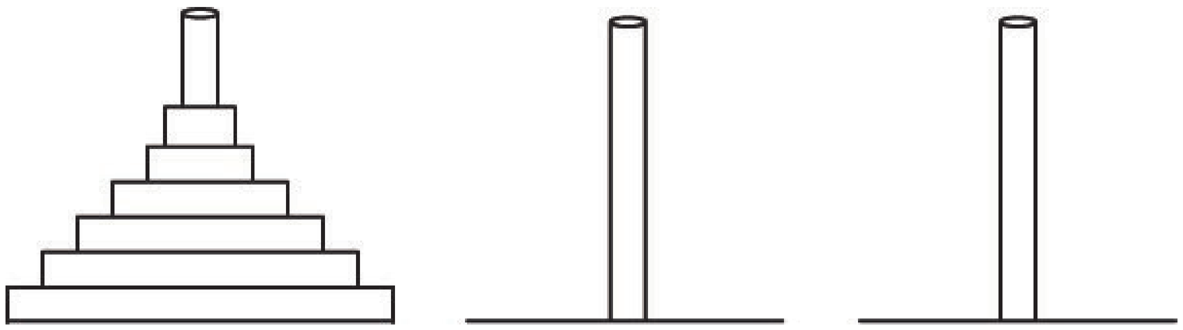

Figure 3.
The Tower of Hanoi puzzle for six disks.

(i) *Death by underpopulation.* Any live cell with fewer than two live neighbors dies.

(ii) *Death by overcrowding.* Any live cell with more than three live neighbors dies.

(iii) *Survival.* A live cell with two or three live neighbors lives on to the next generation.

(iv) *Birth.* Any dead cell with exactly three live neighbors becomes a live cell.

Depending on the initial configuration of life cells, the cells form various patterns—some of which are quite unexpected—throughout the course of the game. For example, the initial cell configuration, called the "glider" (Figure 4, left), descends diagonally one cell down and to the right in four subsequent generations.

Surprisingly, the Game of Life has turned out to have the same computational power as a universal Turing machine: that is, it is theoretically as powerful as any computer with unlimited memory and no time constraints (Berlekamp, Conway, & Guy, 2004, Chapter 25). This has also demonstrated that very complex patterns can emerge from the implementation of a few simple rules and led to the burgeoning area of study of such systems called the cellular automata theory.

Given their ancient history and proliferation of computers in all spheres of human endeavors in the last 50 years, it is surprising that algorithmic puzzles have been recognized as a distinct genre of puzzles only relatively recently. They were identified as such for the first time by A. K. Dewdney in his column in *Scientific American,* where he called them "algopuzzles" (Dewdney, 1987). Many of the puzzles published by Dennis Shasha in his columns in the same publication and *Dr. Dobb's Journal* were certainly algorithmic puzzles; Shasha (2002) called them "cyberpuzzles" in a collection of puzzle-based stories.

Peter Winkler (2004) devoted a special section to algorithmic puzzles in his book of challenging mathematical puzzles. He explicitly used the term "algorithmic puzzles" and described them as puzzles in which a solver is typically presented with a "current situation," a "target state," and a set of "operations" that can be used to modify the situation (p. 77).

A few years later, Dana Richards organized some of Martin Gardner's columns in *Scientific American* in a four-part book (Gardner, 2006), each part covering a broad topic; one of the four was called "Algorithmic Puzzles and Games." In a short introduction to this part of the book, Richards, the book's editor, noted that "a large number of Gardner's problems ask only how to solve a problem, so the puzzle is to devise an algorithm, not to use an algorithm" (p. 227). He included there a very broad range of puzzles, from situational conundrums to matchstick puzzles to chess problems.

Finally, in 2011 Anany and Maria Levitin published a book (Levitin & Levitin, 2011) devoted exclusively to algorithmic puzzles. This collection contains 172 puzzles from very easy to quite hard; most of the puzzles are not new, but they are systematically considered from the algorithm

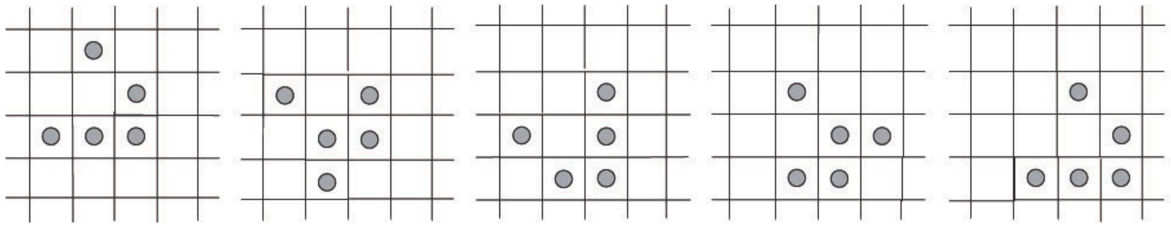

Figure 4.
A "glider" and its four subsequent generations.

design and analysis perspective. The book also contains two tutorials on solving such puzzles.

Since it is natural to consider algorithmic puzzles as particular kinds of mathematical puzzles, we believe that algorithmic puzzles should be well defined (e.g., Robertson, 2017, p. 20). In particular, a solution to an algorithmic puzzle should not depend on a trick or a particular interpretation of the puzzle's statement. Here is an example clarifying this exclusion. Gardner poses the following puzzle in his delightful book *aha! Insight*:

> There are ten glasses in a row: the first five are filled with Kinky Kola, the next five are empty. How many glasses does one need to move to make a row in which the full and empty glasses alternate? (Gardner, 1978, p. 7)

The answer, considered by Gardner as being based on verbal quibble, is 2: pick up the second glass and pour its contents into the seventh, and then pick up the fourth and pour into the ninth. But when Gardner continued with a discussion of the puzzle's general case of an arbitrary even number of glasses, he preferred to discuss the number of glass *switches* to avoid the quibble. It should be admitted, though, that without this quibble the puzzle can hardly require an "Aha!" moment to be solved. A slightly more interesting generalization of this puzzle does not assume that *n* filled glasses are all to the left of *n* empty glasses in a row given (Levitin & Levitin, 2011, #23).

2. CLASSIFICATION OF ALGORITHMIC PUZZLES BY THEIR QUESTION

While there are several ways to classify algorithmic puzzles, the most pertinent one for this paper's subject is a taxonomy based on the question type posed by a puzzle. Here are the main types of such questions:

1. Design an algorithm solving a given puzzle (often in a minimum number of steps).
2. Show that a puzzle has no solution with operations allowed by the puzzle.
3. Find, for a given input, the output of a given algorithm.
4. Find an input yielding a required output of a given algorithm.
5. Find the number of steps made by a given algorithm to solve a puzzle in question.

By far the most common question posed by algorithmic puzzles is of the first type in this taxonomy. This category is broad enough to be subdivided into specific algorithm design strategies (also called "techniques" or "paradigms") used in puzzle solutions:

decrease and conquer	*greedy*
divide and conquer	*dynamic programming*
transform and conquer	*iterative improvement*

These strategies were originally developed for designing algorithms for important problems in computer science. Descriptions of these strategies and examples of their application to solving puzzles can be found in three books (Backhouse, 2011; Levitin, 2012; Levitin & Levitin, 2011) and the paper by Levitin & Papalaskari (2002) advocating a systematic utilization of algorithmic puzzles in teaching algorithms. Of course, a required design strategy is usually not specified in a puzzle statement. In fact, a solver is not assumed to be aware of them, although such knowledge would certainly be very helpful. Further, it is assumed without saying that whenever possible a puzzle should be solved more efficiently than by *exhaustive search* or its variations such as *backtracking* and *branch and bound*—this is why we didn't include them in the above list.

There is one more strategy/heuristic that is used to solve several algorithmic puzzles: *working backwards*. Polya (1957) traced this strategy back to mathematicians of ancient Greece and paraphrased Pappus, who lived around 300 CE, as follows:

> "In analysis, we start from what is required, we take it for granted, and we draw consequences from it, and consequences from the consequences, till we reach a point that we can use as starting point in synthesis. . . . This procedure we call analysis, or solution backwards, or regressive reasoning." (p. 142)

Gardner (2006, Problem 9.8) gave an excellent example of a puzzle solved by working backwards:

> A game of bridge starts with a standard 52-card deck dealt clockwise one card at a time by one of the four players sitting in a circle. A telephone call interrupts a player dealing the cards. When the player returns to the table, no one can remember where he had dealt the last card. Without learning the number of cards in any of the four partly dealt hands, or the number of cards yet to be dealt, how can the player continue to deal accurately, everyone getting exactly the same cards they would have had if the deal had not been interrupted?

Other examples of algorithmic puzzles based on working backwards include Collating the Coins (Schuh, 1968, pp. 17–19), Crowning the Checkers (Gardner, 2006, Problem 10.4), Circle of Zeros and Ones (Nogin, 2014, p. 69), and Trapping the Knight (Hess, 2009, #58). The last of these puzzles, along with Gardner's Interrupted Bridge Game problem, is included in the puzzle sample we provide below as a potentially useful material for insight problem solving investigations.

The second type of algorithmic puzzles are those that have no solution. Typically, such puzzles are solved by finding an *invariant*, a property that is preserved by any operation allowed by the puzzle. If such a property holds for a puzzle's input (initial state) but fails for its output (final state), the puzzle has no algorithmic solution. Two kinds of invariants are encountered more often than others in the algorithmic puzzle universe: coloring and parity. *Coloring* is indispensable for showing impossibility in many checkerboard and checkerboard-like problems. The most well known is the Mutilated Checkerboard puzzle, which asks whether it is possible to tile with 2×1 dominoes a standard 8×8 checkerboard without two diagonally opposite corners; other examples can be found in Golomb's (1994) monograph and in the collection by Levitin and Levitin (2011).

The other frequent invariant type is *even/odd parity* of some aspect of a puzzle. It can be the size of a board to be tiled, some number to be produced, and so on. The most famous are the parity requirements for vertex degrees in a graph to make possible a unicursal traversal of the graph's edges. It solves not only the famous Königsberg›s Bridges puzzle mentioned above in Section 1 but also figure-tracing puzzles and some others (e.g., Levitin & Levitin, 2011). The other famous puzzle solved by the parity argument is the Fifteen puzzle, with the initial configuration of the tiles numbered sequentially from 1 to 15 except for the last two tiles, which are in reverse order (Slocum & Sonneveld, 2006). The puzzle's objective of having all the tiles ordered sequentially is impossible to achieve from this configuration due to the permutation parity argument (e.g., Levitin & Levitin, 2011, #145).

We should also mention that invariant ideas are sometimes used to prove that every sequence of operations reaches the target in the same number of steps or with the same result. The most well-known example is Breaking a Chocolate Bar (e.g., Winkler, 2004, p. 82): find a minimum number of breaks needed to break an $m×n$ rectangular chocolate bar into its constituent squares if you can pick up one piece and break it along any of its vertical or horizontal lines. Another example is Pile Splitting (e.g., Levitin & Levitin, 2011, #104a). Since there are very few puzzles of this kind, we have decided not to create a special category for them.

Puzzles that ask for an output of a given algorithm are relatively rare but do include one of the most famous: Fibonacci's Rabbits problem, mentioned already in Section 1. Another example of such a puzzle is the Locker Doors problem to determine which doors will be open after n passes along a row of n initially closed lockers if on the ith pass (i = 1, 2, . . . , n) every ith door is open if it was closed and closed if it was open (e.g., Levitin & Levitin, 2011, #79).

The Josephus Problem, named after the famous historian of the first century Flavius Josephus, provides an excellent example of puzzles that ask to find an input yielding a desired output of a specified algorithm:

> During the Jewish-Roman war, he [Josephus] was among a band of 41 Jewish rebels trapped in a cave by the Romans. Preferring suicide to capture, the rebels decided to form a circle and, proceeding around it, to kill every third remaining person until no one was left. But Josephus, along with an unindicted co-conspirator, wanted none of this suicide nonsense; so he [Josephus] quickly calculated where he and his friend should stand in the vicious circle. (Graham, Knuth, & Patashnik, 1989, p. 8)

Two other examples are Conway's Solitaire Army (Berlekamp, Conway, & Guy, 2004, pp. 821–823; Levitin & Levitin, 2011, #132) and the Monkey and the Coconuts (e.g., Levitin & Levitin, 2011, #102). Questions about an initial configuration that produces a desired result in the above-mentioned Conway's Game of Life (e.g., Berlekamp, Conway, & Guy, 2004, Chapter 25; Elran, 2012) fall into this category as well.

The last category in this taxonomy consists of puzzles asking to analyze the number of steps executed by a given algorithm. Three very simple puzzles that have often been used in insight problem solving research—Socks, Frog in a Well, and Water Lilies (e.g., Dow & Mayer, 2004; Weisberg, 1995)—are examples of such algorithmic puzzles. The question about the total number of decimal digits needed to consecutively number an n-page book (e.g., Levitin & Levitin, 2011, #19) and Penny Distribution Machine (Levitin & Levitin, 2011, #120) provide others. Although step counting is central to the analysis of algorithms in computer science, there are relatively few puzzles that fall into this category.

3. CLASSIFICATION OF ALGORITHMIC PUZZLES BY INPUT GENERALITY

The other taxonomy of algorithmic puzzles we want to mention here distinguishes between puzzles stated in their general form (e.g., find a lighter fake among $n > 1$ identical-looking coins) as opposed to puzzles stated for a particular instance (e.g., find a lighter fake among eight identical-looking coins). Although computer scientists and mathematicians prefer, as a rule, the general form of algorithmic puzzle statements, there are several reasons for specific instances as well. First, a particular instance can be a traditional form of a puzzle, even though the puzzle in question allows for a natural generalization. For example, many checkerboard puzzles are often stated for a standard 8×8 board.

Second, there are a few algorithmic puzzles that are usually stated for specific small sizes because solutions to those smaller instances contain a main idea behind the puzzle in question, whereas solutions to larger instances can be easily obtained once the small instance is solved. A good example of such a puzzle is the famous Nine Dots problem (Maier, 1930):

> Given a 3×3 square of nine points in the upright square lattice, connect all of them by four straight lines without lifting a pencil from the paper and without redrawing any parts of the lines.[2]

[2] Some cognitive science researchers prefer to ask for crossing nine dots of some physical size with just three rather than four lines, which requires an additional insight (Adams, 1974, p. 17; Batchelder & Alexander, 2012, Problem 3.6).

The puzzle can be generalized to $n \times n$ points for any $n \geq 3$, which can be crossed by $2n - 2$ straight lines, but once the idea of going "outside the box" is perceived for $n = 3$, solving the puzzle's instances for larger values of n poses little difficulty (Levitin & Levitin, 2011, #114).

Third, it might even happen that only a small instance of a puzzle is really interesting. For example, Guarini's Puzzle (e.g., Gardner, 1978, p. 36) about exchanging positions of four knights, two white and two black, located at the corners of a 3×3 chessboard can be generalized to an $n \times n$ board for any $n \geq 3$, but its solutions for $n > 3$ are different and less interesting than for $n = 3$. Also, larger instances of a puzzle may have no solutions at all (e.g., the classic Jealous Husbands puzzle for more than three couples and a two-person boat) or have much more difficult solutions (e.g., Bridge Crossing at Night, Levitin & Levitin, 2011, #7; Rote, 2002).

Finally, casual puzzle solvers usually feel more comfortable with particular instances of puzzles than with their most general statements possible. This is true even though particular numbers in a puzzle statement might complicate the solver's task by forcing the solver to decide whether the solution hinges on some property of the numbers given. Consider, for example, the Lighter False Coin puzzle:

> Identify a lighter fake among $n > 1$ identical-looking coins using a minimum number of weighings on a balance scale without weights.

This puzzle is often posed for $n = 8$. Since $8 = 2^3$, a solver may be led to believe that splitting the coins into two halves on three consecutive weighings is an optimal solution, which is incorrect: the problem can be solved in just two weighings starting with two groups of 3 coins each. For another example, we can again mention the Socks puzzle:

> If you have black socks and brown socks in your drawer, mixed in the ratio of 4:5, how many socks will you have to take out to be sure of having a pair of the same color? (Sternberg & Davidson, 1982, p. 42)

It is easier to solve the more general version of having $n > 2$ socks of two colors, because the ratio given has no impact on the puzzle's solution, making it harder to find. These are excellent examples of the general observation about problem solving made by Polya (1957), who called it *inventor's paradox*: "the more general problem may be easier to solve" (p. 121).

Unlike mathematicians and computer scientists who are typically interested just in puzzle solutions, cognitive scientists are more concerned about explaining cognitive mechanisms behind the solving processes. It might well be that it is harder to study cognitive processes involved in solving general versions of algorithmic puzzles than those involved in solving their particular instances. Further, cognitive science researchers can't expect typical subjects of their experiments to be familiar with algorithms well enough to solve algorithmic puzzles in their general versions. To alleviate this problem, a researcher could just pose a puzzle with a reasonably large particular value of its input: for example, Lighter False Coin for 100 coins. Then, just a number given as an answer should give the experimenter a firm clue whether the solver got the right idea about the optimal algorithm for the problem: for each weighing, divide the remaining coins into *three* groups of as equal size as possible.

We would also like to point out that for some puzzles that can be stated for different values of their input size, a crucial insight can be obtained by considering one or two small instances. This is, of course, an application of *specialization* heuristic, defined by Polya (1957) as "passing from the given set of objects to that of a smaller set, or just one object, contained in the given set" (p. 190). Below are a few examples, starting with a simple river-crossing problem:

> A detachment of n soldiers must cross a wide and deep river with no bridge in sight. They notice two 12-year-old boys playing in a rowboat by the shore. The boat is so tiny, however, that it can only hold two boys or one soldier. How can the soldiers get across the river and leave the boys in joint possession of the boat? (Kordemsky, 1992, #10)

This puzzle can be all but solved by considering just the case of $n = 1$.

A crucial insight about impossibility to solve the Mutilated Checkerboard problem for a standard 8×8 board could be deduced from considering its instances for $n = 2$ and $n = 4$.

The standard version of the Cheap Necklace problem asks to join four 3-link chains into one closed 12-link necklace by spending just 15 cents if the costs of opening and closing 1 link are 2 and 3 cents, respectively. One can speculate that its smaller version, with just three 2-link chains and the total allowed cost of 10 cents, contains a good hint for solving the standard version: the very small number of reasonable alternatives in the former includes the right solution of breaking one of the given chains into individual links, which is harder to see for the larger version. In fact, it would be interesting to compare the usefulness of this hint with those used by Chu, Dewald, and Chronicle (2007) in their experiments with solving the Cheap Necklace problem by students.

In a similar vein, the Eight Coins puzzle asks to move two out of eight coins arranged in two offset rows of four coins so that each coin touches exactly three others (Ormerod, MacGregor, & Chronicle, 2002; Öllinger, Jones, Faber, & Knoblich, 2013). Considering its smaller instance to move one of the four given coins in two offset rows of two coins so that each of the coins touch exactly three others might well lead a solver to the central insight into the problem's solution: the necessity of placing one of the coins on top of the three others.

For the final example of inverting a triangle of 10 coins (Metcalfe, 1986; Chronicle, MacGregor, & Ormerod, 2004), considering the smaller triangles of 3 and 6 coins gives all the hints one might wish for the case of 10 coins.

The usefulness of solving smaller instances of an algorithmic puzzle is, of course, not always a panacea and for some puzzles may even support a wrong idea about the puzzles' solutions. Thus, for the above-mentioned Lighter False Coin puzzle, getting the correct answers for all the cases of $2 \leq n \leq 7$ by always dividing the remaining coins into two equal-size groups (after setting one coin aside if their number is odd) might lead a solver to the wrong conclusion that this is an optimal algorithm for solving the problem for all values of n.

4. ALGORITHMIC PUZZLES AND RESEARCH IN INSIGHT PROBLEM SOLVING

The Tower of Hanoi has proved to be by far the most important algorithmic puzzle for the development of cognitive science. According to Varma (2006), it was called "the drosophila of cognition" by Herbert Simon. Although Varma found this assertion to be an overstatement, he still called this puzzle "a signature task of problem solving that has found wide application in domains such as working memory, intelligence, executive function, and frontal lobe function" (p. 5); he also cataloged such applications in his PhD dissertation devoted to this problem. It is important to note that researchers in modern cognitive science typically don't ask subjects of their experiments to design an algorithm for solving the Tower of Hanoi for an arbitrary number of disks. (Admittedly, it would be an extremely tough task for all but exceptionally gifted persons or computer science majors.) Rather, they either ask subjects to solve the puzzle for a typically small number of disks or prescribe a particular procedure to follow, even allowing some training in its execution. While such setups might well be quite appropriate for getting empirical data about cognitive processes involved in solving specific instances of this puzzle, it is entirely different from designing a general algorithm for it.

Other algorithmic puzzles that have been used often in cognitive research of problem solving include the river crossing puzzle called Missionaries and Cannibals or Hobbits and Orcs (e.g., Guthrie, Vallée-Tourangeau, Vallée-Tourangeau, & Howard, 2015); Water Jars, which asks to get a prescribed amount of water using three jars of given capacities (starting with the pioneering work by Luchins [1942] on the mental set effect); the Fifteen puzzle and its extensions (e.g., Pizlo & Li, 2005); and a few puzzles employed in the difficult research area dealing with insight. We will concentrate on the latter in the remaining portion of this section.

"Insight problems" are often described as problems whose solution involves an "Aha!" experience: a sudden discovery of the solution to a problem that until that point had left the solver baffled (Chu & MacGregor, 2011). Not surprisingly, puzzles of different kinds have been used as examples of such problems (Sternberg & Davidson, 1995, Part II). Somewhat surprisingly, just a small number of puzzles, called "classic" by Chu and Macgregor (2011), have dominated the literature on insight problem solving. They include the above-mentioned Nine Dots, Cheap Necklace, Eight Coins, Ten-Coin Triangle, Mutilated Checkerboard, Socks, and Water Lilies, all of which are instances of algorithmic puzzles, along with some other verbal, spatial, and mathematical brainteasers (see, e.g., Dow & Mayer, 2004; Weisberg, 1995). Newer additions to this limited repertoire include matchstick arithmetic with Roman numerals (Knöblich, Ohlsson, Haider, & Rhenius, 1999), compound remote associates (CRAs) (Bowden, Jung-Beeman, Fleck, & Kounious, 2005), and rebus puzzles (MacGregor & Cunningham, 2008). Benefits of this expansion of the insight problem universe notwithstanding, it ought to be noted that CRAs and rebus puzzles, along with other verbal problems such as anagrams and brainteasers based on alternative meanings of words in their statements, share the same weakness: their solutions require a native-speaker command of a language used. Without special care to account for this, experimental results involving language-dependent puzzles offered to a diverse student body can be distorted in an unpredictable fashion. Algorithmic puzzles do not share this weakness. In addition, the flexibility of varying an input of an algorithmic puzzle might provide the experimenter with a set of similar problems of different difficulty—a desirable feature for insight research (Goldstone & Pizlo, 2009).

Despite a superficial clarity of what insight problems are, as introduced above, a more careful analysis of this notion has revealed many uncertainties (e.g., Chu & Macgregor, 2011; Danek, Wiley, & Öllinger, 2016; Dominowski & Dallob, 1995; Mayer, 1995). Weisberg (1995) suggested considering a problem an insight problem if its solution involves discontinuity in thinking and restructuring understood as a change in the solver's representation of the problem. According to his taxonomy, if solving a problem involves both of these elements, it is an *insight problem*; otherwise, it is not even if its solution may involve an "Aha!" experience. Further, if restructuring is the only way to solve the problem, it is said to be a *pure insight problem*; if solving a problem involves discontinuity but it can be solved both with and without restructuring, it is said to be a *hybrid insight problem*. Ash, Cushen, and Wiley (2009), on the other hand, argued that a better approach would be to identify insight problems specifically by their obstacles. Recently, Weisberg (2015) suggested a more holistic model of the problem-solving process in which the traditional insight sequence of an impasse followed by a restructuring and an "Aha!" solution is just one possible aspect of the process; the model was partly based on the empirical data reported by Fleck and Weisberg (2013) and its analysis by the authors.

We would like to conclude this section by making two more observations about insight problems. The first, limited to algorithmic puzzles, is an obvious fact that such a puzzle can be an insight problem for some of its instances but not for the others. For example, the Frying Pancakes puzzle (see Problem 8 in Section 5.1 below) is an insight problem for any odd $n > 1$ pancakes and trivial for all the other values of n.

The second observation concerns a potential difference between an insight needed to find a solution and an insight needed to justify the solution's correctness. Consider, as a simple example, the problem of placing the largest number of kings on an 8×8 chessboard so that no 2 kings threaten each other—that is, no 2 kings are on adjacent squares vertically, horizontally, or diagonally. One can hardly expect any discontinuity in thinking from a reasonable solver before he or she places 16 kings on the board (e.g., 4 kings per row in four nonadjacent rows). On the other hand, proving that it is indeed the maximum number of kings possible does require a nontrivial restructuring to represent the board as a union of four 4×4 squares in each of which no more than 1 king can be placed (e.g., Levitin & Levitin, 2011, p. 16).

5. FUTURE WORK

In this section, we first list 12 algorithmic puzzles that, in our view, could be useful for future research of insight problem solving. Then we suggest several experiments involving other issues: solving puzzles by brute force and working backwards, transfer questions, and a board coloring impact.

5.1 TWELVE ALGORITHMIC PUZZLES FOR INSIGHT RESEARCH

Since several researchers have advocated for a wider range of problems used in studying insight (e.g., Batchelder & Alexander, 2012; Chu & Macgregor, 2011), it seems logical to consider which algorithmic puzzles beyond the few classics could be useful in insight research. For a sample of such puzzles given below, we sought puzzles that satisfy the following requirements:

- A puzzle's solution must not be obvious to the solvers, but it should be simple enough for at least some non–computer science majors to get it in a realistic amount of time.
- A possibility of solving a puzzle by trial and error or by labor-intensive computations should be minimized to the degree possible.
- A puzzle should be well defined; in particular, its wording should contain no ambiguities, such as the meaning of the word "move" in the Alternating Glasses puzzle discussed above.
- A puzzle should be stated in a way—e.g., for a particular input size—that preserves an underlying algorithmic idea without requiring a complete algorithm description in its general form.
- Whenever there is a meaningful difference in solutions for different input values or a puzzle's difficulty levels, several versions of the puzzle can be given.

Solutions to all the puzzles in the sample, along with brief comments, are given in the appendix.

Problem 1. A crazy king of some unfortunate country ordered all 100,000 adults in his kingdom to participate in a single-elimination tournament to determine the best player in a game the king had invented. (The game could not end in a tie, and as is the case for any single-elimination tournament, every losing player is immediately eliminated from subsequent rounds of play.) How many games must his subjects play to determine a winner?

Problem 2. A little girl counts from 1 to 1,000 using the fingers of her left hand as follows. She starts by calling the thumb 1, the first finger 2, the middle finger 3, the ring finger 4, and the little finger 5. Then she reverses direction, calling the ring finger 6, the middle finger 7, the first finger 8, and the thumb 9, after which she calls the first finger 10, and so on. If she continues to count in this manner, on which finger will she stop?

Problem 3. Is there a way for a chess knight to start at the lower left corner of a standard 8×8 chessboard, visit all the squares of the board exactly once, and end at the upper right corner? (The knight's moves are L-shaped jumps: two squares horizontally or vertically followed by one square in the perpendicular direction.)

Problem 4. Can one transform the left table in Figure 5 into the right table by a sequence of steps if on every step one can either exchange two rows or exchange two columns of the current table?

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

12	10	9	11
16	14	5	13
8	6	7	15
4	2	3	1

Figure 5.
Initial table (left) and target table (right).

Problem 5. Four people find themselves on the same side of a river they need to cross using a rickety footbridge. It is dark, and they have one flashlight. A maximum of two people can cross the bridge at one time. Any party that crosses, either one or two people, must have the flashlight with them. The flashlight must be walked back and forth; it cannot be thrown, for example. Person 1 takes 1 minute to cross the bridge, person 2 takes 2 minutes, person 3 takes 5 minutes, and person 4 takes 10 minutes. A pair must walk together at the rate of the slower person's pace. For example, if person 1 and person 4 walk together, it will take them 10 minutes to get to the other side. If person 4 returns the flashlight, a total of 20 minutes has passed. Can they cross the bridge in 17 minutes?

Problem 6. You have $n > 2$ identical-looking coins and a two-pan balance scale with no weights. One of the coins is a fake, but you don't know whether it is lighter or heavier than the genuine coins, which all weigh the same. How can you determine whether the fake coin is lighter or heavier than the others with just two weighings? Solve the problem for (a) $n = 11$ and (b) $n = 10$.

Problem 7. Interrupted Bridge Game mentioned above in Section 2.

Problem 8. You need to make pancakes using a skillet that can hold only two pancakes at a time. Each pancake should be fried on both sides; frying one side of a pancake takes 1 minute, regardless of how many pancakes are fried at the same time. What is the minimum amount of time you will need to make 13 pancakes?

Problem 9. You have a rectangular chocolate bar marked into 5×6 squares, which you wish to break into 30 constituent squares. At each step, you can pick up one piece and break it along any of its marked horizontal or vertical lines. What is the minimum number of such steps you will need?

Problem 10. There are seven glasses on the table, all standing upside down. In one move, you can turn over any four. Is it possible to turn all the glasses up by a sequence of such moves?

Problem 11. What is the minimum number of moves on an infinite chessboard needed for a knight to reach a position from which it can move only to a previously visited square?

Problem 12. Dissect a square into n smaller squares for (a) $n = 7$; (b) $n = 8$; (c) $n = 9$.

5.2 FEW EXPERIMENTS BASED ON ALGORITHMIC PUZZLES

In addition to the puzzles suggested above for insight research, we would like to suggest the following experiments with human solving of algorithmic puzzles. At least for some of these experiments, it could also be interesting to investigate a possible dependence of the observed results on subjects' psychometric measures of intelligence (e.g., Burns, Lee, & Vickers, 2006).

Experiment 1. Ask a group of subjects to solve the Lighter False Coin puzzle for $n = 8$ coins. Ask the subjects after they turn in their solutions whether they considered any alternatives to the solution they provided and if not why.

The common incorrect answer to this puzzle, reported to be given during job interviews, is 3, with the first weighing of two four-coin subsets. The fact that many solvers are not bothered with proving the correctness of their answers is noteworthy but not very surprising. What is surprising and, in our view, more important is the implication that human solvers are reluctant to solve puzzles by exhaustive search even when it is not explicitly forbidden and the number of possible alternatives is very small. What could force solvers to consider the alternatives is to pose this puzzle with the requirement that it must be solved in two weighings. This hypothesis can be checked by asking a control group of subjects to solve this version of the puzzle.

One can also ask another group of subjects to solve the same problem for $n = 9$ coins and compare the results with those for $n = 8$. (Results for $n = 9$ could be better, because this value might suggest a division of the coins into three groups more readily than for $n = 8$, although this advantage could be overwritten by the wrong insight that one coin can be set aside before the first weighing.)

Experiment 2. To investigate whether a solver always tries to solve a puzzle by an obvious brute-force approach, one can use impossible-tiling puzzles such as Questionable Tiling (e.g., Levitin & Levitin, 2011, #12) that asks whether it is possible to tile an 8×8 board with dominoes so that no two dominoes form a 2×2 square and an even simpler question about tiling this board with Z-tetrominoes, which are tiles made of four unit squares glued together in the shape of the letter "Z" (e.g., Levitin & Levitin, 2011, #38e). Both puzzles can be solved by the same brute-force attempt to produce a required tiling, which might be more difficult for more sophisticated solvers who may try to find an invariant providing an impossibility proof.

Experiment 3. Investigate an impact of considering smaller instances in solving algorithmic puzzles (by either giving subjects a hint of desirability of such considerations or providing specific small instances or doing both for two different groups of subjects to compare the effectiveness of these two methods). We mentioned in Section 3 several puzzles that can be used for such investigations: Mutilated Checkerboard, Cheap Necklace, Eight Coins, and Ten-Coin Triangle. Also, one can use, among others, two of the insight problems mentioned above: Row and Column Exchanges (Problem 4) with a 2×2 table as a smaller instance and Trapping the Knight (Problem 11) with an 8×8 board as a natural smaller instance. It would certainly also be useful to ask subjects in a control group that is provided with neither hints nor specific smaller instances whether they considered smaller instances on their own and compare the performance data of those who did with those who did not. Our hypothesis is that considering smaller instances of a given problem is not a standard tool of a typical problem solver and hence must be taught, but our hypothesis needs an empirical verification.

Experiment 4. Investigate how easy it is for human solvers to transfer a standard application of an algorithm to a nonstandard one. For the experiment suggested below, the algorithm in question is binary search, which is based on repeatedly decreasing the problem's size by half. It should be familiar to many if not most subjects from the game Twenty Questions. We suggest comparing subjects' performance in solving the following three puzzles:

(a) What is the minimum number of questions with yes/no truthful answers needed to guarantee "guessing" an integer between 1 and 100, inclusive? Indicate a way to achieve this. (This standard application of binary search requires seven questions in the worst case.)

(b) What is the minimum number of questions with yes/no truthful answers needed to guarantee "guessing" a selected card in a randomly shuffled 52-card deck laid out in 4 rows of 13 cards each? Indicate a way to achieve this. (Applying binary search either to the card suits and values from 1 for an ace to 13 for a king or to the row and column numbers of this two-dimensional array requires 2 + 4 questions in the worst case.)

(c) Forty index cards with different English words written on them (one word per card) lay before you and your friend on a table arranged in 4 rows and 10 columns. What is the minimum number of questions with yes/no truthful answers needed to guarantee "guessing" a word selected by your friend? Indicate a way to achieve this. (Applying binary search to the row and column numbers of this two-dimensional array requires 2 + 4 questions in the worst case. Less elegantly, one can apply binary search to the 40 cards numbered from 1 to 40, say, row by row from the leftmost to the rightmost card, although such a numbering would have to be communicated in the questions asked.)

It could be useful for analyzing results of this experiment to ask the experiment's subjects after they turn in their solutions whether they have been familiar with the game of Twenty Questions and analyze the results separately for the subgroup that has and the subgroup that has not been.

Experiment 5. Most puzzles that must be solved backwards are not easy for human solvers. It is not usually clear, however, what makes it so: the fact that solvers do not consider this heuristic to begin with or because such puzzles remain difficult even after a solver is told that they need to be solved backwards. To get some empirical data regarding this dilemma, we suggest comparing the effectiveness of the hint that a problem should be solved by working backwards for, say, such puzzles as Interrupted Bridge Game and Trapping the Knight mentioned above in Section 2. Are the puzzles easily solved after the hint was given, or do they remain difficult even after that?

Experiment 6. Investigate the impact of a board's colorings on solving puzzles about a path through a given board. In particular, one can compare success rates and solution speeds in finding a path through the squares of the three boards in Figure 6a (see next page) and proving that such a path does not exist for the three boards in Figure 6b (see next page). (A path in question may proceed through any sequence of horizontally or vertically adjacent squares and is not required to return to its starting square.)

The checkerboard coloring should be helpful for finding a path for the board in Figure 6a. Since the number of light squares there is more by 1 than the number of dark squares, a path through all the squares must start and end on light squares. One such path, which exploits the board's symmetry, is given in Levitin and Levitin (2011, p. 114). The checkerboard coloring is also helpful in proving that there exists no path for the board in Figure 6b. Indeed, the colors of squares in such a path would have to alternate, whereas the number of black squares is larger than the number of white squares by 3. As to the success rates in solving the puzzles for the boards with row coloring, one would expect them to be comparable to or worse than those for the uncolored boards.

As another version of the board coloring, one can investigate a random coloring of the board's squares in two or more colors. Also, in addition to comparing an impact of the board colorings for each of the two boards, one can compare the observed data for the similarly colored boards to ascertain relative difficulty of these two problems, the first of which has a solution and the second of which does not.

6. CONCLUSION

Algorithmic puzzles constitute a relatively small but important class of puzzles. Their nature puts them in a unique position of being of interest to mathematicians, computer scientists, and cognitive science researchers, making some of these puzzles a fruitful research topic. We reviewed major milestones in the long history of algorithmic puzzles and discussed two ways to classify them: by a question posed and by the generality of their input.

The Tower of Hanoi, one of the most widely known algorithmic puzzles, has been used extensively in different areas of cognitive science. Just a few instances of algorithmic puzzles have become a staple of research in insight problem solving. In our view, many more algorithmic puzzles can be used for that purpose. As specific examples, we suggested 12 algorithmic puzzles that, in our opinion, could be useful for experiments in this challenging research area. In fact, one can claim that the previous classifications of insight problems into three categories of verbal, special, and mathematical by Dow and Mayer (2004) or four categories of brainteasers and riddles, geometric, manipulative, and mathematical by Weisberg (1995) should be expanded by adding the class of algorithmic puzzles. The main advantages of algorithmic puzzles over other kinds of insight problems are their language independence and flexibility of varying instance sizes. Finally, in addition to expanding the universe of problems for insight research, we proposed several experiments dealing with other cognitive aspects of solving algorithmic puzzles.

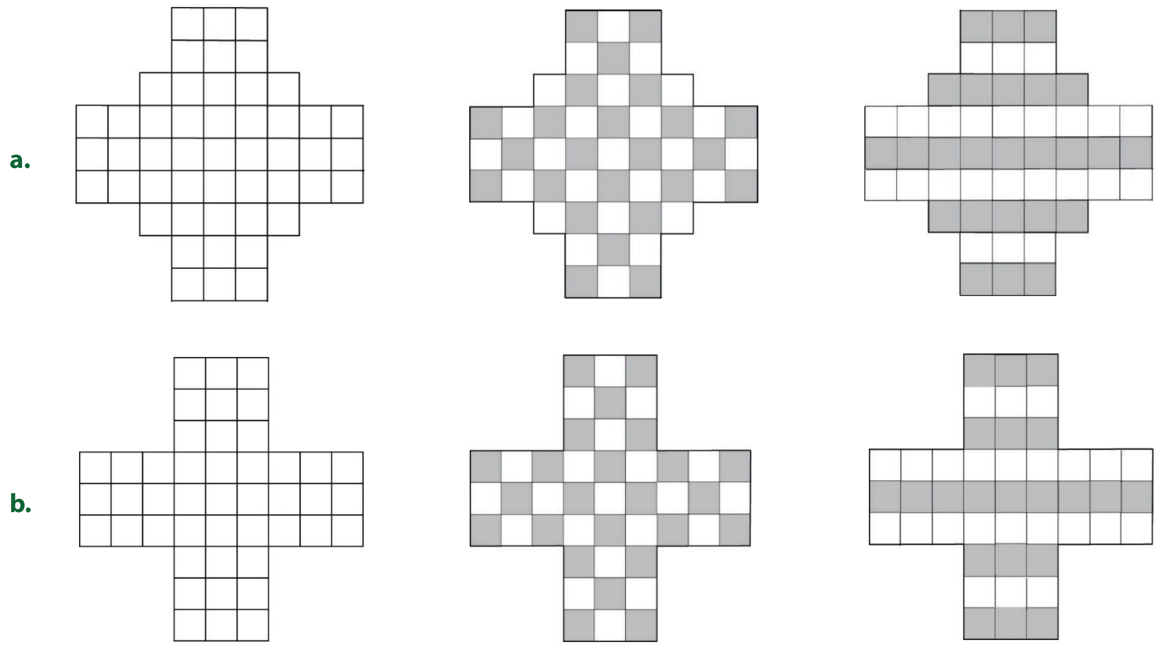

Figure 6.
Colored versions of two boards (a. and b.) to find a path through their squares or prove that such a path does not exist.

REFERENCES

Adams, J. L. (1974). *Conceptual blockbusting*. San Francisco: W. H. Freeman.

Ascher, M. (1990). A river-crossing problem in cross-cultural perspective. *Mathematics Magazine, 63*(1), 26–29. https://doi.org/10.2307/2691506

Ash, I. K., Cushen, P. J., & Wiley, J. (2009). Obstacles in investigating the role of restructuring in insightful problem solving. *Journal of Problem Solving, 2*(2), 3–41. https://doi.org/10.7771/1932-6246.1056

Backhouse, R. (2011). *Algorithmic problem solving*. Chichester, West Sussex, UK: Wiley.

Batchelder, W. H., & Alexander, G. E. (2012). Insight problem solving: A critical examination of a formal theory. *Journal of Problem Solving, 5*(1), 56–100. https://doi.org/10.7771/1932-6246.1143

Berlekamp, E. R., Conway, J. H., & Guy, R. K. (2004). *Winning ways for your mathematical plays, Vol. 4* (2nd ed.). Wellesley, MA: A K Peters.

Bowden, E. M., Jung-Beeman, M., Fleck, J., & Kounios, J. (2005). New approaches to demystifying insight. *Trends in Cognitive Sciences, 9*(7), 322–328. https://doi.org/10.1016/j.tics.2005.05.012

Bullmore, E., & Sporns, O. (2009). Complex brain networks: Graph theoretical analysis of structural and functional systems. *Nature Reviews Neuroscience*, *10*(3), 186–198. https://doi.org/10.1038/nrn2575

Burns, N. R., Lee, M. D., & Vickers, D. (2006). Are individual differences in performance on perceptual and cognitive optimization problems determined by general intelligence? *Journal of Problem Solving, 1*(1), 5–19. https://doi.org/10.7771/1932-6246.1003

Chronicle, E. P., MacGregor, J. N., & Ormerod, T. C. (2004). What makes an insight problem? The roles of heuristics, goal conception, and solution recoding in knowledge-lean problems. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, *30*(1), 14–27. https://doi.org/10.1037/0278-7393.30.1.14

Chu, Y., Dewald, A. D., & Chronicle, E. P. (2007). Theory-driven hints in the cheap necklace problem: A preliminary investigation. *Journal of Problem Solving, 1*(2), 18–32. https://doi.org/10.7771/1932-6246.1010

Chu, Y., & MacGregor, J. N. (2011). Human performance on insight problem solving: A review. *Journal of Problem Solving, 3*(2), 119–150. https://doi.org/10.7771/1932-6246.1094

Danek, A. H., Wiley, J., & Öllinger, M. (2016). Solving classical insight problems without aha! experience: 9 dot, 8 coin, and matchstick arithmetic problems. *Journal of Problem Solving, 9*(1), 47–57. https://doi.org/10.7771/1932-6246.1183

Dewdney, A. K. (1987). Algopuzzles: Wherein trains of thought follow algorithmic tracks to solutions. “Computer Recreations” column. *Scientific American*, *256*(6), 128–131. https://doi.org/10.1038/scientificamerican0687-128

Dominowski, R. L., & Dallob, P. (1995). Insight and problem solving. In R. J. Sternberg & J. E. Davidson (Eds.), *The nature of insight* (pp. 33–62). Cambridge, MA: MIT Press.

Dow, G. T., & Mayer, R. E. (2004). Teaching students to solve insight problems: Evidence for domain specificity in creativity training. *Creativity Research Journal, 16*(4), 389–398. https://doi.org/10.1080/10400410409534550

Dudeney, H. E. (1967). *536 Puzzles & Curious Problems*. New York: Scribner.

Elran, Y. (2012). Retrolife and the pawns neighbors. *College Mathematics Journal, 43*(2), 148–152. https://doi.org/10.5948/UPO9781614448013.029

Fleck, J. I., & Weisberg, R. W. (2013). Insight versus analysis: Evidence for diverse methods in problem solving. *Journal of Cognitive Psychology, 25*(4), 436–463. https://doi.org/10.1080/20445911.2013.779248

Gardner, M. (1978). *aha! Insight*. New York: W. H. Freeman.

Gardner, M. (2006). *The colossal book of short puzzles and problems*. New York: Norton.

Garey, M. R., & Johnson, D. S. (1979). *Computers and intractability: A guide to the theory of NP-completeness*. New York: W. H. Freeman.

Goldstone, R. L., & Pizlo, Z. (2009). New perspectives on human problem solving. *Journal of Problem Solving, 2*(2), 1–5. https://doi.org/10.7771/1932-6246.1055

Golomb, S. W. (1994). *Polyominoes: Puzzles, patterns, problems, and packings* (2nd ed.). Princeton, NJ: Princeton University Press.

Graham, R. L., Knuth, D. E., & Patashnik, O. (1989). *Concrete mathematics: A foundation for computer science*. Reading, MA: Addison-Wesley.

Guthrie, L. G., Vallée-Tourangeau, F., Vallée-Tourangeau, G., & Howard, C. (2015). Learning and interactivity in solving a transformation problem. *Memory & Cognition*, *43*(5), 723–735. https://doi.org/10.3758/s13421-015-0504-8

Hess, D. (2009). *All-star mathlete puzzles*. New York: Sterling.

Knöblich, G., Ohlsson, S., Haider, H., & Rhenius, D. (1999). Constraint relaxation and chunk decomposition in insight problem solving. *Journal of Experimental Psychology: Learning, Memory, and Cognition, 25*(6), 1534–1555. https://doi.org/10.1037/0278-7393.25.6.1534

Knott, R. (2017, May). *Ron Knott's multimedia web site on the Fibonacci numbers, the Golden section and the Golden string*. Retrieved from http://www.maths.surrey.ac.uk/hosted-sites/R.Knott/Fibonacci/fib.html

Kordemsky, B. A. (1992). *The Moscow puzzles: 359 mathematical recreations*. New York: Dover.

Levitin, A. (2012). *Introduction to the design and analysis of algorithms* (3rd ed.). Boston: Pearson/Addison-Wesley.

Levitin, A., & Levitin, M. (2011). *Algorithmic puzzles*. New York: Oxford University Press.

Levitin, A., & Papalaskari, M. A. (2002). Using puzzles in teaching algorithms. *ACM SIGCSE Bulletin, 34*(1), pp. 292–296. https://doi.org/10.1145/563517.563456

Luchins, A. S. (1942). Mechanization in problem solving: The effect of Einstellung. *Psychological Monographs*, *54*(248), i–95. https://doi.org/10.1037/h0093502

MacGregor, J. N., & Cunningham, J. B. (2008). Rebus puzzles as insight problems. *Behavior Research Methods, 40*(1), 263–268. https://doi.org/10.3758/BRM.40.1.263

Maier, N. R. F. (1930). Reasoning in humans: On direction. *Journal of Comparative Psychology*, *10*(2), 115–143. https://doi.org/10.1037/h0073232

Mayer, R. E. (1995). The search for insight: Grappling with Gestalt psychology's unanswered questions. In R. J. Sternberg & J. E. Davidson (Eds.), *The nature of insight* (pp. 3–32). Cambridge, MA: MIT Press.

Metcalfe, J. (1986). Premonitions of insight predict impending error. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 12, 623–634. https://doi.org/10.1037/0278-7393.12.4.623

Nogin, M. (2014). *Strategies of Problem Solving* (2nd ed.). Fresno: California State University.

Öllinger, M., Jones, G., Faber A. H., & Knoblich G. (2013). Cognitive mechanisms of insight: The role of heuristics and representational change in solving the eight-coin problem. *Journal of Experimental Psychology: Learning, Memory, and Cognition, 39*(3), 931–939. https://doi.org/10.1037/a0029194

Ormerod, T. C., MacGregor, J. N., & Chronicle, E. P. (2002). Dynamics and constraints in insight problem solving. *Journal of Experimental Psychology: Learning, Memory, and Cognition, 28*(4), 791–799. https://doi.org/10.1037/0278-7393.28.4.791

Petković, M. S. (2009). *Famous puzzles of great mathematicians*. Providence, RI: American Mathematical Society. https://doi.org/10.1090/mbk/063

Pizlo, Z., & Li, Z. (2005). Solving combinatorial problems: The 15-puzzle. *Memory & Cognition*, *33*(6), 1069–1084. https://doi.org/10.3758/BF03193214

Polya, G. (1957). *How to solve it: A new aspect of mathematical method* (2nd ed.). Princeton, NJ: Princeton University Press.

Robertson, S. I. (2017). *Problem solving: Perspective from cognition and neuroscience* (2nd ed.). London: Psychology Press.

Rote, G. (2002). Crossing the bridge at night. *Bulletin of the EATCS, 78*, 241–246.

Schuh, F. (1968). *The master book of mathematical recreations*. New York: Dover.

Shasha, D. E. (2002). *Doctor Ecco's cyberpuzzles: 36 puzzles for hackers and other mathematical detectives*. New York: Norton.

Sigler, L. E. (2002). Fibonacci's *Liber Abaci*: A translation into modern English of Leonardo Pisano's Book of Calculation. New York: Springer. https://doi.org/10.1007/978-1-4613-0079-3

Simon, H. A., & Newell, A. (1971). Human problem solving: The state of the theory in 1970. *American Psychologist*, *26*(2), 145–159. https://doi.org/10.1037/h0030806

Singmaster, D., & Hadley, J. (1992). Problems to sharpen the young: An annotated translation of Propositiones ad acuendos juvenes, the oldest mathematical problem collection in Latin, attributed to Alcuin of York. *Mathematical Gazette, 76*, 102–126. https://doi.org/10.2307/3620384

Slocum, J., & Sonneveld, D. (2006). *The 15 puzzle book: How it drove the world crazy*. Beverly Hills, CA: Slocum Puzzle Foundation.

Sternberg, R. J., & Davidson, J. E. (1982). The mind of the puzzler. *Psychology Today, 16*, 37–44.

Sternberg, R. J., & Davidson, J. E. (Eds.). (1995). *The nature of insight*. Cambridge, MA: MIT Press.

Varma, S. (2006). *Computational model of Tower of Hanoi problem solving.* Doctoral dissertation, Vanderbilt University, Nashville.

Weisberg, R. W. (1995). Prolegomena to theories of insight in problem solving: A taxonomy of problems. In R. J. Sternberg & J. E. Davidson (Eds.), *The nature of insight* (pp. 157–196). Cambridge, MA: MIT Press.

Weisberg, R. W. (2015). Toward an integrated theory of insight in problem solving. *Thinking & Reasoning, 21*(1), 5–39. https://doi.org/10.1080/13546783.2014.886625

Winkler, P. (2004). *Mathematical puzzles: A connoisseur's collection*. Natick, MA: A K Peters.

APPENDIX

This appendix contains solutions to the sample of 12 algorithmic puzzles given in the paper, along with brief comments.

Solution to Problem 1. The total number of matches is equal to 99,999: each match yields 1 loser, and exactly 100,000 - 1 losers need to be produced to get a single winner of the tournament.

The insight here is to concentrate on match losers. Simon and Newell (1971) mentioned the same problem with 109 players and commented on its solution as follows: "There are many 'trick' problems of this kind where a selection of the correct problem space permits the problem to be solved without any search" (p. 154).

The problem's statement doesn't specify explicitly a way of dealing with the initial number of players not being equal to a power of 2, which requires giving some players "byes"—transfers of players directly to the next round because they have no opponent assigned to them. There are two ways to give byes:

i. The byes are given to the fewest players in the first round to have the number of players left for the second round equal to a power of 2.

ii. The byes are given to the fewest players to have an even number of players in each round.

The numbers of byes in these two versions are actually different (see, e.g., Levitin & Levitin, 2011, pp. 190–191), but they are not needed to solve the problem.

Solution to Problem 2. The girl will stop on her first finger. Here is how the finger count starts:

finger	count	count
thumb	1	9
first	2	10
middle	3	11
ring	4	12
little	5	13
ring	6	14
middle	7	15
first	8	16

It is easy to see from the table that in order to answer the question, all one needs is to find the remainder of the division of 1,000 by 8, which is equal to 0. This implies that when the girl reaches 1,000 she will be on her first finger, the same one she will be on while calling any number divisible by 8.

The puzzle is from Martin Gardner's *Colossal Book of Short Puzzles and Problems* (Gardner, 2006, p. 63; see also Levitin & Levitin, 2011, #6). A similar problem was included in Henry Dudeney's puzzle collection (Dudeney, 1967, Problem 164).

Solution to Problem 3. The journey in question is impossible. The squares where the knight starts and ends its move are always of the opposite color. To visit all the squares of the board once, it would need to make 63 moves; since this number is odd, such a journey would need to start and end on squares of the opposite color. But the squares of the lower left and upper right corners of the board are colored the same color, making the journey in question impossible.

The puzzle (e.g., Levitin & Levitin, 2011, #18) is a typical example of exploiting colors of the board's squares as the invariant idea.

Solution to Problem 4. The answer is "no." Row and column exchanges preserve the numbers they contain. This is not the case for the tables given: for example, 5 and 6 are in the same row in the initial table but are in different rows in the target table.

The attractiveness of this impossibility puzzle (Levitin & Levitin, 2011, #5) lies in the fact that the solution's invariant (the puzzle's insight) is neither coloring nor parity.

Solution to Problem 5. The sequence of moves solving the puzzle is shown in Figure 7 (see next page).

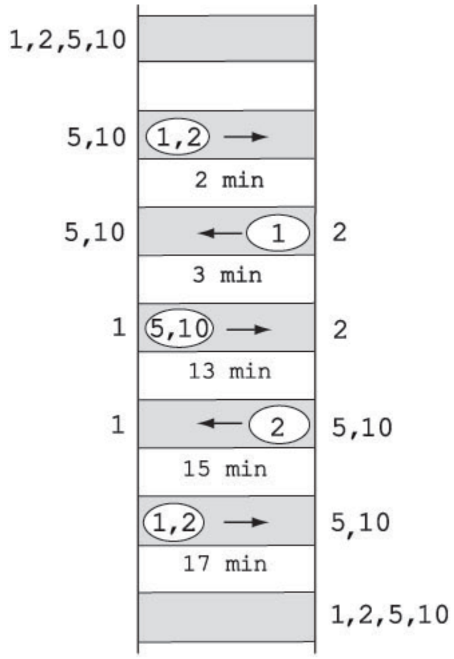

Figure 7.
Solution to Problem 5: Labels 1, 2, 5, and 10 represent the four people, respectively; the arrows indicate the crossing directions.

This relatively recent river-crossing puzzle has proved to be surprisingly difficult for many solvers. For more information, see Levitin and Levitin (2011, p. 87).

Solution to Problem 6. Here is a solution to an arbitrary number of coins $n > 2$. Start by taking aside one coin if n is odd and two coins if n is even. After that, divide the remaining even number of coins into two equal-size groups and put them on the opposite pans of the scale. If they weigh the same all these coins are genuine, and the fake coin is among the coins set aside. So, we can weigh the set-aside group of one or two coins against the same number of genuine coins: if the former weighs less, the fake coin is lighter; otherwise, it is heavier.

If the first weighing does not result in balance, take the lighter group; if the number of coins in it is odd, add to it one of the coins initially set aside (which must be genuine). Divide all these coins into two equal-size groups and weigh them. If they weigh the same all these coins are genuine, and therefore the fake coin is heavier; otherwise, they contain the fake, which is lighter.

What makes this puzzle (e.g., Levitin & Levitin, 2011, #44) different from typical weighing puzzles is its objective, which is not to identify a fake coin but instead only to determine whether it is lighter or heavier than the genuine ones.

Solution to Problem 7. The player deals the bottom card to himself, then continues dealing from the bottom counterclockwise.

This is a remarkable example of solving a problem by working backwards, which constitutes its insight.

Solution to Problem 8. One can make 3 pancakes in 3 minutes as follows. First, fry pancakes 1 and 2 on one side. Then, fry pancake 1 on the second side together with pancake 3 on its first side. Finally, fry both pancakes 2 and 3 on the second side. After making the first three pancakes in this fashion, the remaining 10 pancakes can be fried in pairs, spending the total of 10 minutes to fry them on both sides. Thus, the total time to fry 13 pancakes will be 13 minutes. This is the minimum time possible, because 13 pancakes have 26 sides to be fried, and any algorithm can fry no more than 2 sides in one minute.

The insight for solving this old puzzle is to think not in terms of whole pancakes but rather in terms of their individual sides. This makes it possible to fry 3 pancakes in 3 minutes, which is the most important instance of the puzzle.

Solution to Problem 9. Since each break increases the number of pieces by 1, 29 breaks are needed to solve the problem, and *any* sequence of 29 breaks will do this.

As soon as a solver recognizes the importance of the total number of pieces at hand, the answer becomes obvious. Despite this "obvious" answer, the puzzle has been reported to stump some very high-powered mathematicians (Winkler, 2004, p. 93).

Solution to Problem 10. Since the number of glasses turned over in one move is even, the parity of the number of glasses that are upside down will always remain odd, as it was in the initial position. Hence, the final position in which the number of upside-down glasses is 0, which is even, cannot be reached.

This is a typical impossibility puzzle with a parity-based solution.

Solution to Problem 11. Solving the puzzle backwards, consider the knight in a final position from which it can move only to 1 of the 8 previously visited squares. Since all these squares are of the same color, the knight had to visit at least 1 new square of the opposite color between every pair of them, for a total of 7 such squares. Therefore, the answer is 15, as shown in Figure 8, where the squares are numbered in the order they are visited.

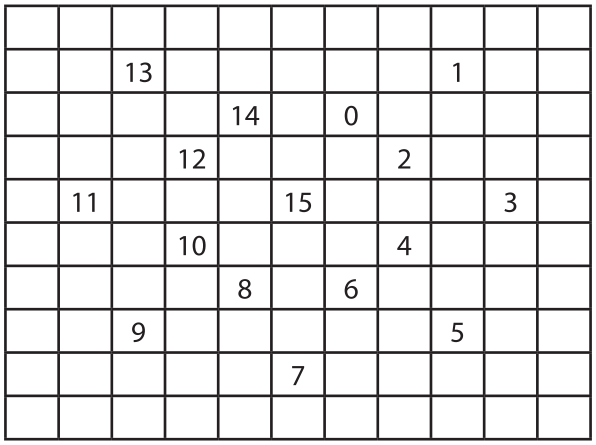

Figure 8.
A tour solving Problem 11, starting at square 0 and ending at square 15.

Solution to Problem 12. Possible dissections of a square into 7, 8, and 9 squares are shown in Figure 9. Once a solver realizes that smaller squares are not required to be of the same size, the case of $n = 7$ becomes almost obvious. The dissection into $n = 8$ smaller squares, which can be generalized to an arbitrary even $n = 2k$, where $k > 1$, is obtained by having $2k - 1$ equal squares along two adjacent sides of the given square, with the side length of each smaller square equal $1/k$-th of the side length of the given square. The dissection into $n = 9$ smaller squares, which can be generalized to an arbitrary odd $n > 5$, i.e., $n = 2k + 1 = 2(k - 1) + 3$, where $k > 2$, is obtained by first dissecting the given square into $2(k - 1)$ squares, as described above for even n's and then dissecting any of the obtained squares (e.g., the one in the top left corner) into four smaller ones.

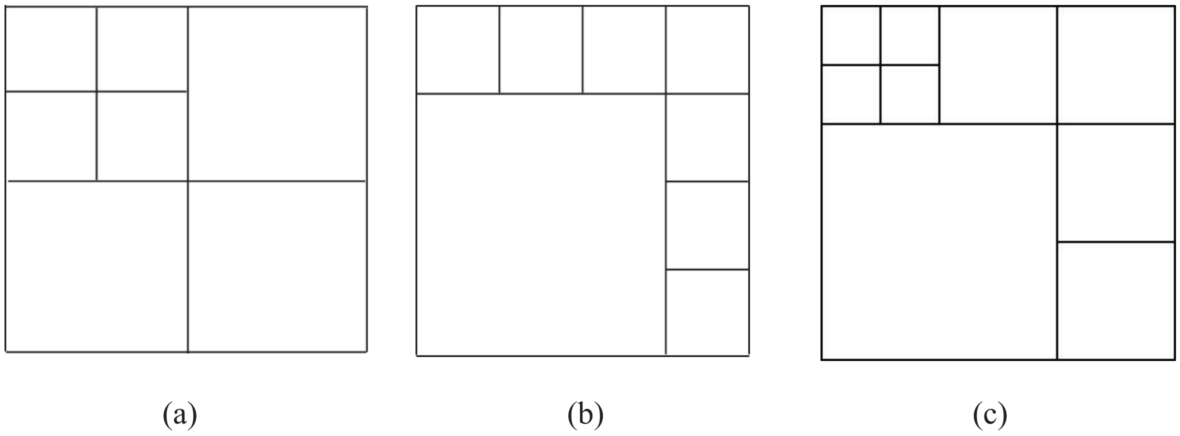

Figure 9.
Square dissection into (a) 7 squares, (b) 8 squares, and (c) 9 squares.