topiCS in Cognitive Science
in press

When is psychology research useful in artificial intelligence? A case for reducing computational complexity in problem solving

Sébastien Hélie
Department of Psychological Sciences, Purdue University

Zygmunt Pizlo
Department of Cognitive Sciences, University of California Irvine

A problem is a situation in which an agent seeks to attain a given goal without knowing how to achieve it. Human problem solving is typically studied as a search in a problem space composed of states (information about the environment) and operators (to move between states). A problem like playing a game of chess has 10^{120} possible states, and a travelling salesperson problem with as little as 82 cities already has more than 10^{120} different tours (similar to chess). Biological neurons are slower than the digital switches in computers. An exhaustive search of the problem space exceeds the capacity of current computers for most interesting problems, and it is fairly clear that humans cannot in their lifetime exhaustively search even small fractions of these problem spaces. Yet, humans play chess and solve logistical problems of similar complexity on a daily basis. Even for simple problems humans do not typically engage in exploring even a small fraction of the problem space. This begs the question: How do humans solve problems on a daily basis in a fast and efficient way? Recent work suggests that humans build a problem representation and solve the represented problem – not the problem that is out there. The problem representation that is built, and the process used to solve it, are constrained by limits of cognitive capacity and a cost–benefit analysis discounting effort and reward. In this article, we argue that better understanding how humans represent and solve problems using heuristics can help inform how simpler algorithms and representations can be used in artificial intelligence to lower computational complexity, reduce computation time, and facilitate real–time computation in complex problem solving.

Introduction

A problem is a situation in which an agent seeks to attain a given goal without knowing how to achieve it. Example problems include winning at tic–tac–toe or winning a battle, air traffic control, control of an uninhabited vehicle, getting to checkmate in chess, visually–guided navigation, proving a logic theorem, solving math and physics problems, cracking the enigma code, or formulating a new scientific theory. Some problems are more visual, such as planning a tour around a grocery store, while others are more abstract, such as proving a theorem using predicate logic. Newell and Simon's (1972) formulation of problem solving as a goal–directed search in a problem space is still a largely–accepted view. According to that view, the problem space is composed of a number of states and operators. The states can be defined either explicitly or implicitly and correspond to all the problem information available to the problem solver. The solver begins in an initial state, and aims to reach a goal state. All other problem states are called intermediate states. The solver moves from state to state using a set of operators. For example, the initial state in a game of tic–tac–toe is an empty grid, and a goal state has three identical markers aligned horizontally, vertically, or diagonally. All other configurations of markings are intermediate states, and the operator is simply to put a mark in one of the empty squares in the grid.

Not all problems are as simple as tic–tac–toe: It contains only one clearly defined operator (putting a marker) and 26,830 possible states (i.e., configurations of markers). However, problem complexity can increase quickly. A problem like playing a game of chess has 10^{120} possible states, and choosing what food to buy at a grocery store with as little as 1,000 different items can produce more than 10^{300} possible lists of items (states) (Choi, Kim, & Lee, 2021). It is important to note at this point that biological neurons are slower than the digital switches in a modern computer. An exhaustive search of problem spaces exceeds the capacity of current digital computers for most realistically interesting problems, and it is fairly clear that humans cannot in their lifetime exhaustively search even small fractions of these problem spaces. Yet, humans play chess and solve logistical problems of similar complexity on a daily basis. Even for a simple problem such as tic–tac–toe, typical players do

* Corresponding author: Dr. Sébastien Hélie (shelie@purdue.edu), 703 Third Street, West Lafayette, IN 47907 USA. This research was supported in part by NSF grant #1662230 to SH. The authors would like to thank Zixuan (Sofia) Zhao for helpful discussions and suggestions.

not engage in exploring even a small fraction of the problem space. This begs the question: How do humans solve problems on a daily basis in a fast and efficient way?

In this article, we argue that studying human problem solving can provide important insight about how problems can be represented and simplified to allow for reaching solutions more efficiently. Given the limited cognitive resources available to humans, simplification of the problem space or operators is a necessity. While artificial intelligence (AI) does not suffer from these same cognitive limitations, simplifying the problems presented to AI agents the same way humans are simplifying problems can allow for more efficient and timely solution to complex problems.[1] The remainder of this article is organized as follows. First, we provide a short introduction to the notion of growth in computational complexity. This allows for determining the number of operations needed to search a problem space and solve a given problem. Next, we review how humans reduce computational complexity by either using heuristic operators or simpler representations. This presentation is followed by recommendations of how the knowledge obtained by studying human problem solving can be used to improve problem solving by AI agents. We conclude with general considerations and future directions.

Computational complexity in problem solving

The most common way to represent problem solving scenarios is to specify a graph in which one or more nodes are starting states (e.g., all the information about the environment[2] available before beginning to solve the problem), one or more nodes are goal states (e.g., all the information about the environment available after solving the problem), and edges represent transitions between pairs of states (operators) (Fleischer, Hélie, & Pizlo, 2018; Newell & Simon, 1972). Each edge has a cost. In some puzzles like 15–puzzle (the common tile game where one needs to order 15 tiles on a 4×4 grid) and Tower of Hanoi (where one needs to move disks from one peg to another while maintaining stacking order), each edge has a cost of 1 and represents a step. In problems such as the travelling salesperson problem (TSP), the cost of an edge is the length of the shortest path between pairs of cities. In the previous shopping list example, the cost would be a negative function of the utility of adding a product to the list. An agent, human or a computer, has to figure out how to get to the goal state from the start state. Any path would represent a solution, but the shortest (least–cost) path is the optimal solution. Typically, solving a problem assumes search in the graph. During the search, an agent performs computations. The computational complexity of a search algorithm solving a problem is based on how many computations are required to produce an optimal solution. In a Big–O notation, what is important is the amount of computations in the worst case scenario. More precisely, the Big–O notation states how quickly the number of computations grows with a problem size. Typical growth functions include logarithmic, linear, polynomial (e.g., quadratic or cubic), and exponential. The previous growth functions are listed in decreasing order of desirability, and the first three are generally considered "tractable" (Rich, Blokpoel, de Haan, & van Rooij, 2020). However, exponential growth quickly outgrows computational resources and problems that can only be solved by exponentially growing algorithms are generally considered "intractable".

The TSP is an example problem where finding the optimal solution requires a number of computations that grows exponentially. Yet, the TSP problem does not appear to human subjects as extremely difficult. When presented with a 20–city TSP, a subject is likely to produce a tour that is very close to optimal, and the tour will be produced in about one minute (Dry, Lee, Vickers, & Hughes, 2006; Graham, Joshi, & Pizlo, 2000). When a 60–city TSP is used, the subjects produce a near–optimal tour in 3 minutes. So, on average, the time the human subjects use in producing TSP tours is proportional to the number N of cities – suggesting that humans use an algorithm with linear growth. One possibility is that subjects are solving a different problem (Carruthers, Stege, & Masson, 2018). This is hinted by the observation that humans are not producing THE optimal solution, but instead a reasonably good suboptimal solution. Another possibility is that the subjects are solving a TSP problem, which asks for the shortest tour, but the algorithm that the human is using prevents them from producing such tours. The mental operations are limited by the size of working memory (WM), which can only hold and manipulate a few items at a time. A 20–city TSP has $19!/2$ tours, which is about 10^{17}. The number of neurons in the brain is one million times smaller than this number. As a result, physical limitations in humans likely prevent the subject from exploring even a fraction of the possible tours. With a WM that can hold only a couple items at a time, the subject cannot perform any substantial amount of search because the number of options at any given step must be no more than WM capacity. AI algorithms do not generally have to operate under such constraints. With humans, WM is also not the only constraint. Time is another one. If the subject evaluated one tour per second, evaluating all tours in a 20–city TSP would take the time between the Big Bang and today. A human subject produces a TSP tour in 60 seconds, and there is reason to believe that most of this

[1]AI suffers from its own set of limitations, e.g., limited access to meaning (Searle, 1980). However, these limitations are outside the scope of this article. Nevertheless, it is important to note that no intelligent system, natural or artificial, operates without limitation.

[2]In this article, we use environment as defined in the reinforcement learning literature (Sutton & Barto, 1998). As a result, the environment includes everything that is outside the agent (e.g., perceptual cues, rewards, other agents, etc.).

time is spent moving a computer mouse and clicking on the 20 points. The actual solution time is likely to be less than one minute. The next section discusses how humans achieve high performance in complex problems despite these limitations.

Human problem solving

Given the high complexity of most interesting problems, how do humans solve problems? In the previous section, we briefly reviewed cases where humans were solving complex problems (e.g., the TSP, grocery shopping) almost instantaneously. How is this achieved given the biological and cognitive limitations? Not only is the number of neurons and synapses in the brain "small" when compared to the number of operations needed to implement the algorithms required to solve many problems, but neurons are also notoriously slow. The maximum firing rate of biological neurons is about 1000 Hz, and typical firing rates are less than 100 Hz. This is orders of magnitude slower than the digital switches in a modern computer. An exhaustive search of problem spaces exceeds the capacity of current digital computers for most realistically interesting problems, and it is fairly clear that humans cannot in their lifetime exhaustively search even small fractions of these problem spaces.

In addition to biological limitations, humans and other animals have important cognitive limitations. For example, WM capacity estimates typically range from 4–7 items for young adults (Ashby, Ell, Valentin, & Casale, 2005). These estimates provide an upper bound on the number of states or possibilities that can be considered simultaneously. In addition to space limitations, there are also temporal limitations. For example, the well–known attentional blink phenomenon shows that it takes time to engage and disengage attention to target stimuli (Raymond, Shapiro, & Arnell, 1992). Given these limitations, human problem solvers need to reduce the computational complexity of the algorithms they use to find satisfactory problem solutions (Carruthers et al., 2018). This "over–performance" of humans in problem solving is likely a consequence of using *heuristics* and *re–representation*. The next two subsections discusses these topics in turns.

Searching for better operators: Heuristics in problem solving

There are at least two different systems that people use to make decisions: an analytical process and a non–analytical process. The former is logical and cost–based, while the second is fast and intuitive (Miller & Geraci, 2016; Mueller & Dunlosky, 2017). AI agents typically implement the first (analytical) process. As a case in point, "learning" in AI typically refers to finding the optimal parameters of an objective function (Abel, 2003; Newell & Simon, 1972). However, unlike AI systems, human decision is often biased or influenced by changing the surface representation of the problem. Humans tend to use different available cues to generate an intuitive guess (the second, non–analytical, system). This process is called *heuristic*, and it can be either accurate or inaccurate, depending on the situation (Castel, 2008; Jia et al., 2016; Serra & Ariel, 2014).

Because humans use heuristics, the selection of cues is an important aspect of the problem solving process. Relevant cues can be extracted from the problem space based on their form, association, availability, or other properties. For example, recognition heuristics give more weight to familiar objects: A familiar ham is more likely to be selected as an edible object compared to a green egg (Gigerenzer, 2000). In most cases, when the cue aligns with its utility in the problem, the bias can lead to fast and accurate judgments (Kornell, Rhodes, Castel, & Tauber, 2011). However, humans can be distracted by irrelevant cues, such as font and color (Mueller, Dunlosky, Tauber, & Rhodes, 2014; Mueller & Dunlosky, 2017). Weighing irrelevant cues can result in wasting resources and inaccurate beliefs. For example, Mueller and Dunlosky (2017) instructed participants that blue stimuli are calmer and more easily processed by human eyes. After receiving these (false) instructions, participants gave a higher judgment of learning score to blue stimuli when the actual perceptual fluency difference between the colors should not have a significant effect on judgments of learning (Mueller & Dunlosky, 2017).

Note that the tendency to attend to irrelevant cues can also facilitate problem solving in some cases: One can arrange the irrelevant cues artificially to guide the participants towards using relevant cues. For example, Rouinfar, Agra, Larson, Rebello, and Loschky (2014) asked participants to solve physics problems. The problems were represented by diagrams, but participants could ask for cues. The cue was a shape that would appear somewhere in the diagram. While the shape was irrelevant to solving the problem, its location in the diagram would correspond to an area of the diagram that was useful in solving the problem. The appearance of the shape shifted the participant's attention to that part of the diagram and increased the likelihood of solving the problem. As can be seen from these previous results, cues and heuristics influence human problem solving processes, and these can be beneficial in directing searches and solving problems.

Participants also tend to use different heuristics voluntarily to make rational and fast decisions. Humans can extract important properties in the problem space to simplify complicated problems and generate standardized procedures, such as decision trees, to guide and ensure efficient problem solving in the future (Gigerenzer, 2000). While participants aim at rationality whenever possible, constraints of WM resources make the notion of rationality bounded (Simon, 1972) and effort costs need to be taken into consideration. As a result, the most exhaustive methods do not always yield the most rational result. Humans use various heuristic building blocks to control the evidence searching process. For example, consider a simple problem where participants need to make a choice between two competing alternatives. Many such questions are common in everyday decisions (e.g., should I order the soup or salad). Simple fast and frugal heuristics can be very useful in these situations. Fast and frugal heuristics take only part of the evidence (in the limit one piece of information) when making comparisons between the two alternatives. The information considered can be based on validity, availability, or randomness. Performance with these heuristics improve when the cue(s) is (are) valid and had a high success rate in the past. When only one piece of information is used the heuristic is called Take the best (Gigerenzer, 2000). Humans tend to implement the take the best strategy in solving problems. For example, Yahosseini and Moussaïd (2019) asked participants to search for rewards from a designed landscape. The reward followed deterministic cues. The results show that participants could efficiently detect the deterministic cue, and locate rewards efficiently based on the cue. Participants also gave priority to novel solutions and exploited solutions with immediate feedback (Gardner, 2019).

While focusing on a subset of information may be rational in some cases, a more balanced exploration and evaluation of joint cues can be suitable for more complicated problems (Korf & Felner, 2002). Elimination heuristics (as used in Yahosseini & Moussaïd, 2019) are suitable for cases where there exist many cues and alternatives, and considering all cases is too costly (or effortful). Limiting the amount of information considered can reduce the dimensionality of the problem, but alternatives are ruled out (Gigerenzer, 2000). If the correct alternative is ruled–out by the heuristics then performance will be suboptimal.

Searching for a better problem representation: Creative problem solving

Asides from using simpler operators and heuristics, changing the problem space is also a useful tool to simplify problems. This process is often called *re–representation* and has been studied mostly in the context of creative problem solving (Hélie & Olteteanu, in press; Hélie & Sun, 2010). In creative problem solving, changes in the problem representation are often achieved through the use of analogies (Gentner, 1983) and metaphors (Lakoff & Johnson, 1999). Metaphors and analogies have been shown to facilitate representational change, restructuring, and representational redescription (MacGregor & Cunningham, 2009; Ohlsson, 1984), all topics that have recently gathered renewed interest (Olteteanu & Indurkhya, 2019). However, not all re–representation requires using imagery. For example, Schooler, Ohlsson, and Brooks (1993) used the following problem:

Water lilies double in area every 24 hr. At the beginning of the summer, there is one water lily on the lake. It takes 60 days for the lake to become completely covered with water lilies. On which day is the lake half–covered?

Participants typically begin by attempting to write down an equation that describes the growth of water lilies. Many participants fail because of the recursive nature of the problem. However, once the participant notices that if the number of lilies doubles every day, and the lake is full on day 60, this implies that on day 59 the lake is half–covered. On the next day (day 60), the number of lilies double and the lake is fully covered. In this case, shifting the problem representation (e.g., starting with a full lake instead of an empty one) essentially amounts to solving the problem, and one does not need to write down any equation to solve it.

While the previous example is typically referred as an *insight* problem, not all examples of change in problem space involve creativity. Consider the TSP on a Euclidean plane. As a reminder, the TSP refers to a task of finding the shortest tour of N cities (points). An algorithm that guarantees finding the shortest tour may, in the worst case, have to perform a number of operations that is an exponential function of the number of cities (Lawler, Lenstra, Rinnooy Kan, & Shmoys, 1992). This is impractical even for moderate N because an exponential function grows very quickly. Yet, human participants can find near–optimal tours in linear time for N as large as 120 (Dry et al., 2006). Consider Figure 1, which shows four clusters of cities. This 13–city TSP has over 200 million possible tours, but it usually takes less than half a minute for a human observer to produce a near–optimal tour. The four clusters form a convex quadrilateral, which determines the order in which these clusters should be visited. The remaining decisions are about the order in which the cities within the clusters should be visited. It is not obvious what the best order is, but selecting an incorrect order within clusters does not matter much. Errors within clusters only produce small differences in the overall TSP tour length. The hypothesis that humans produce TSP tours by forming and using clusters was originally proposed by Graham et al. (2000).

Two things are worth noting at this point. First, the approach described to solve the TSP is very methodical and does not involve insight or creativity (i.e., there is no *Aha!* moment associated with the use of clustering). Second, the representation of the problem that humans solve is different from the problem that is posed to them (Fleischer et al., 2018). In the TSP, participants are asked to find the shortest tour, which requires calculating the lengths of all possible tours. However, humans instead decompose the global optimization problem into a series of local optimizations that can be handled with limited WM. Local optimizations, in general, may produce highly suboptimal solutions. But participants avoid bad solutions by using global–to–local series of optimizations: global optimization on a coarse represen-

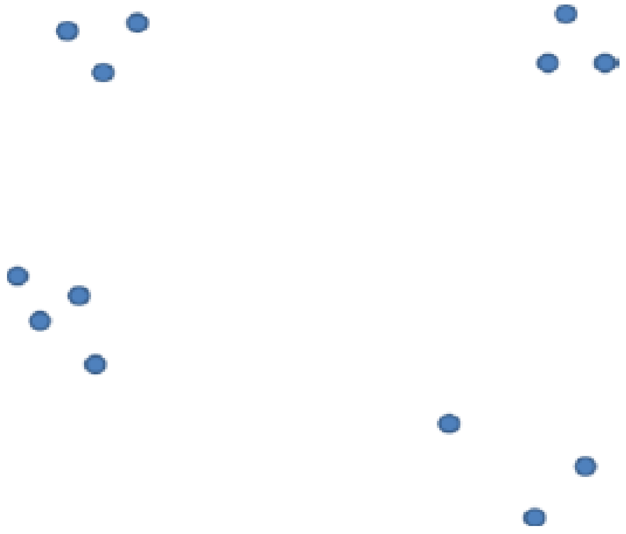

Figure 1. TSP with 13 cities. The clusters form a quadrilateral.

tation to link clusters, followed by within–cluster tours on gradually finer–grained representations. This works well if the problem space allows for hierarchical clustering based on some proximity metric and the assumption that the same metric can be applied everywhere in the problem space (a form of symmetry of the clustering operation). When the TSP becomes non–metric however, human participants struggle because they can no longer produce good tours using local computations (Sajedinia, Pizlo, & Hélie, 2019).

Some problems are visual, like TSP on a Euclidean plane (described above) or visual navigation, but other problems may not have an obvious visual representation (Lovett & Forbus, 2017). Algebra problems, first order logic, and chess are examples. Visual problems have a useful iconic representation, but more abstract problems require the creation of a propositional representation (Kunda, McGregor, & Goel, 2013). One common type of propositional representation is directed graphs (Angerer & Schreiber, 2019). Propositional representations can emerge from restructuring (or recoding) the iconic visual representation, or be created following task instructions (e.g., learning the rules and constraints of the problem). There is currently no agreement or integrative theory of how humans build internal representations. But regardless of how the problem representation was created, qualitatively different representations can be processed using different operations (Hélie & Sun, 2010). For example, multiplying roman numerals is tedious, but multiplying arabic numbers is much easier.

What does this all mean? We argue that one of the fundamental characteristics of human problem solving is that humans use, whenever possible, the concept of *direction* in solving problems (i.e., they move towards the solution without considering alternative choices; Pizlo & Li, 2005). When they are asked to produce a shortest path, which is equivalent to the minimization of distance traveled, they actually do not use distance. Using distance (or cost), as is common in AI algorithms, will necessarily lead to expensive search. Using direction allows humans to decide about the next step, without considering alternatives. This may sound like magic, but its not. Consider the simplest case of 2 points on a Euclidean plane. When asked to connect these points by a shortest curve, humans draw a straight line that begins at one of these points and has a direction pointing to the other point. When drawing the straight line, humans are not aware of how long it is. The length is irrelevant here for obvious reasons. The same applies when one needs to walk to someone to transfer an object. If the transaction is pressing, the individual will try to find the quickest way to reach the other party. Distance is not measured; one just walks towards the other person and feels confident that s/he travelled the shortest distance.

When a surface other than a plane is considered, the concept of a straight line generalizes to the concept of a geodesic in differential geometry (Hilbert & Cohn-Vossen, 1952). A geodesic line is the shortest path between two points and it can be produced by maintaining the same direction when moving from point to point. In other words, a geodesic on a surface does not introduce an unnecessary curvature to the path. When graphs are used to represent problems, the concept of a geodesic on a smooth surface does not apply. Straight lines don't exist in graphs and neither curvatures of surfaces or paths. But, if there is a way to translate the global characteristics of the problem into a local decision for where to go next, this condition resembles direction on a Euclidean plane. Using direction is a simple way to produce near–optimal solutions to a wide range of problems very quickly with minimal use of WM. This is true with the TSP, 15–puzzle, and a number of other problems. The visual system may provide the computational machinery for handling spatially–global computations effectively, including the use of global features in specifying local directions. This is possible because the visual system is a massively parallel computational system. This feature allows the visual system to keep multiple representations of the problem and to navigate the representations from a global–to–local or local–to–global characteristics.

Reducing computational complexity *is* intelligence

Why is human research in problem solving important for AI development? Let's begin by using a simple example problem: 15–puzzle (see Figure 2). It has been established that for the 15–puzzle, the length of the optimal solution ranges from 0 to 80 (depending on how the tiles are arranged in the start state). For comparison, the length of the optimal solution for the simpler (similar) 8–puzzle ranges from

0 to 31. For larger sizes of this puzzle ($n^2 - 1$), the length of optimal solutions are not known, yet. The search space is simply too large. Korf (1985) proposed an algorithm for solving 15–puzzle called "iterative deepening A*". The algorithm guarantees finding the shortest path, but the number of nodes that are evaluated can be quite large. To illustrate the algorithm's performance, Korf randomly generated 100 start states of 15–puzzle and solved them using the algorithm. The optimal solutions had lengths between 41 and 66 steps.

Consider the start state shown in the left side of Figure 2. In Korfs (1985) example problems, the goal state had the empty square in top left corner, and all other digits were ordered from left–to–right and from top–to–bottom (see the right side of Figure 2). The empty square was coded using the number 0 (whereas in Figure 2, the empty square is simply represented by an empty space). If we write the 4 rows of 15–puzzle as one row, Korfs goal state becomes: {0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15}. The start state shown in Figure 2 is: {15 2 12 11 14 13 9 5 1 3 8 7 0 10 6 4}. Iterative deepening A* uses a cost function that counts for each tile how many squares it needs to be moved to reach its goal state. The cost for all tiles are then summed. For example, Tile 2 in Figure 2 needs to be moved one step to the right to reach the goal state. In contrast, Tile 15 needs to be moved 6 steps (3 steps down, 3 steps right). Using this cost function, the start state shown in Figure 2 has a cost of 43, which is a lower bound of the shortest path. The optimal solution for this problem requires 65 steps. Korfs algorithm visited 6,009,130,748 states to determine the shortest path to get to the goal state. The total number of states in 15–puzzle is about 10^{13}. This means that Korfs algorithm explored close to 0.001 of the entire problem space. This is a large fraction of the problem space. Humans will never do that. One of us (Pizlo) rearranged the tiles from Korfs start state to the goal state and it took 121 steps (about 2 minutes). It is clear that Pizlo did not find the shortest path (65 steps), but his effort (121 steps) was much more economical than the effort of Korfs AI algorithm (over 6 billion steps). To put succinctly, "producing the optimal solution" is not the same as "optimally producing a solution". Typically, AI agents focus on the former, whereas humans focus on the latter.

The example above illustrates that humans solve problems by both changing the problem representation (Pizlo did not consider all tiles simultaneously) and repeatedly applying simple (cheap) operators. For example, most humans solving the 15–puzzle focus on one tile at a time, and treat all other tiles as indistinguishable. This simplification greatly reduces the size of the problem space. The assumption here is that applying the operator (moving a tile) is cheap when only the position of the tile being moved is considered, so 121 applications of the operator in a reduced problem space may be less costly than applying the operator 65 times in the complete problem space (when the positions of all tiles are considered).

We argue that solving problems with AI could take a similar approach. Generating human–like operators and agents to facilitate relevant processes is very important in the field of robotic and artificial intelligence. While an AI agent could, in theory, have infinite memory and processing power, computation is expensive in time and energy. AI developers have already begun replicating many important humans cognitive processes, such as vision, language processing, etc. These implementations enable artificial agents to understand the environment in a more human–like way (Barto, Singh, & Chentanez, 2004; Wang et al., 2019). Just like humans, one could program an artificial agent that focuses on one tile at a time and ignore all other tiles. The memory requirement and branching of the search would be much reduced, so even if more operations are needed it may still be faster and more economical. This can be important for real–time computation – e.g., with a moving robot or when the environment is dynamic.

While the previous example reduces computation by changing the problem representation, the operators are not changed: humans are still moving tiles the same way as they would when considering all tile locations. It can be quite clear that in some other cases the operators that are allowable can be changed by the problem representation. For example, one can use localist (one node = one concept) or distributed representations (one pattern of activation = one concept) in connectionist networks. Localist representations are symbolic and do not allow for a notion of distance or direction in hyperspace (all the vectors are orthogonal).[3] In contrast, distributed representations define a space with distances and directions. Hence, the type of approach proposed earlier where direction is used to solve spatial problems cannot be used with localist representations (because concepts are symbolic and either the same or different). The way the problem is represented can affect the operations that are possible. Heuristics that require ordering distances or proximities would not be possible with a purely symbolic representation, and these are important when the quality of a solution is to be evaluated continuously (more or less correct) instead of all–or–none (correct or incorrect).

What is the best way to include these heuristics in AI? Humans use heuristics spontaneously out of necessity: our cognitive abilities and the amount of effort we are whiling to expend are limited. It is very likely that evolution played an important role in selecting agents who could optimize the cost/benefit trade–off of the algorithms they used to solve problems. Computers do not face these limitations and are not the result of millions of years of evolution, so these heuristics need to be either hard–coded in the agent, or an

[3]One can allow for real–value activation of localist representations, giving a notion of match, but there is still no real geometry between the concepts.

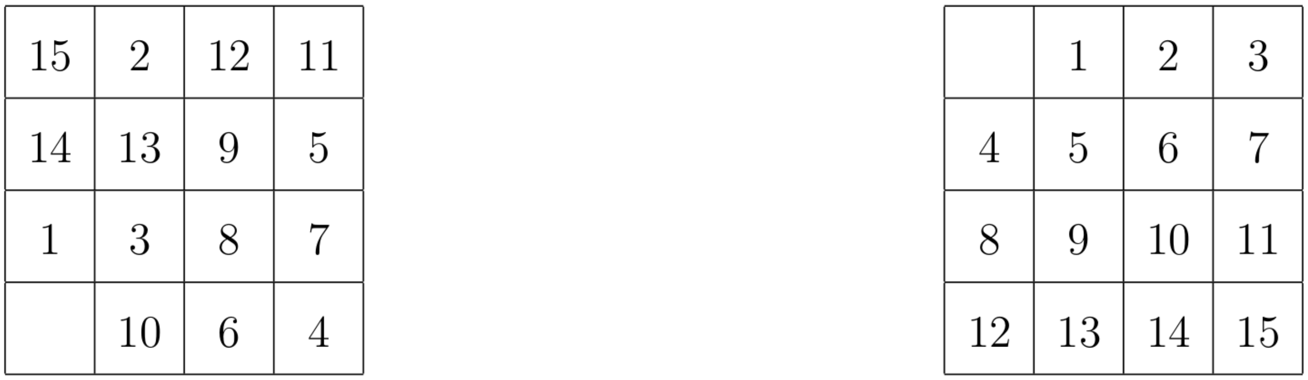

Figure 2. The 15–puzzle. *Left:* Start state of Korf's (1985) Problem 88. *Right:* Goal state.

automated processes to learn heuristics needs to be included. A mixture of both is probably ideal, because the mechanisms required to solve the problems, which depend on the problem space representation itself, can be simple or complicated depending on the problem representation. Given that most natural problems are ill–defined, the problem information needs to be abstracted by detecting the patterns and the rules from the environments (Davidson & Sternberg, 2003). In humans, this can be achieved by contextualizing the problem, that is, bringing to mind knowledge about past situations that were similar using, e.g., analogies, metaphors, or case–base reasoning.

The way the problem space is represented restricts what type of operators are allowed to search the space (Newell & Simon, 1972). Capturing and approximating human heuristics by integrating different cues can enable computer agents to generate better human–like solutions when facing complicated problems (Gardner, 2019). Reinforcement learning agents are common computational algorithms that can implement human–like heuristics (e.g., Katsikopoulos, Şimşek, Buckmann, & Gigerenzer, 2021; Şimşek, 2020). They model a sequential process where humans interact with the environment repeatedly and generate valuable solutions based on feedback from the environment (Barto et al., 2004). As a result, they capture the dynamic of the environment directly. Reinforcement learning agents have been used to solve difficult problems such as the Rubiks cube (Agostinelli, McAleer, Shmakov, & Baldi, 2019), the gizmo problem (Dandurand, Bowen, & Shultz, 2004), etc. The agents achieve good performance in performing the tasks, typically by using cues to guide the search in useful ways. The cue weights are learned using trial–and–error. The human problem solving process, however, is more complicated. Humans use previous knowledge to solve new problems, and they tend to learn efficiently from examples. Dandurand et al. (2004) defined imitative learning to capture analogy, reasoning, and generalization related abilities. Imitative learning could be very beneficial, if successfully applied by the artificial agent, in reducing training effort and enabling expert–like thinking.

Integrating different evidence and making human–like decisions are important aspects of reinforcement learning agents. However, this process can be extremely hard given the number and dimension of pieces of information, alternatives, and cues available in the environment (Shah & Oppenheimer, 2008). Human's natural tendencies and heuristics to simplify the environment are not directly available in computer agents. They have been developed over years of cultural evolution (Rich et al., 2020). They can be suboptimal, but they can also be useful in different situations: effort reduction is usually assumed to be the main purpose of using heuristics. For example, perceptual fluency and preference lead to a fast and frugal thinking style, which gives rise to recognition heuristics or warm glow heuristics styles of processing. The biases introduced by these kinds of heuristics change the availability of different cues and enable a more targeted search (Shah & Oppenheimer, 2008). This biased integration of cues according to their perceived importance helps to develop faster and better–structured reinforcement learning agents. It also helps to balance between exploration and exploitation to exploit the most beneficial resources and construct the most effective solutions (Drake, Kheiri, Özcan, & Burke, 2020; Smith, 1983; Yahosseini & Moussaïd, 2019).

Conclusion

This article reviewed findings showing that humans often times do not directly solve the problem that is presented to them but instead solve a simpler version of it. Yet, the solution provided by human solvers is often very good, if not optimal. Many problems are highly complex and cannot be solved with reasonable computing resources in a reasonable amount of time. Studying human problem solving can be informative as to how the computational complexity of algorithms used to solve problems can be reduced, and using simpler algorithms or including heuristics to solve problems in AI agents could be beneficial for fast, efficient, real–time problem solving.

Overall, arguments in this article support the usefulness of using human heuristics in AI programs. However, one needs to also remember that heuristics have remarkable limitations in guiding problem solving. They cannot guarantee efficiency or solutions (Abel, 2003). The right type of heuristics enables the algorithms to solve the problems more effectively, but it does not guarantee an optimal solution ultimately. The trade–off between optimality and efficiency needs to be considered by interested researchers (Gardner, 2019). This is why adaptability and human oversight are needed. Much research suggests that human participants use different strategies or heuristics based on the nature of the problems. The problem representation that is built is critical if the problem space is not intuitive, and human feedback can help AI agents learn better representations and heuristics.

References

Abel, C. F. (2003). Heuristics and problem solving. *New Directions for Teaching and Learning*, *2003*(95), 53–58.

Agostinelli, F., McAleer, S., Shmakov, A., & Baldi, P. (2019). Solving the rubiks cube with deep reinforcement learning and search. *Nature Machine Intelligence*, *1*(8), 356–363.

Angerer, B., & Schreiber, C. (2019). Representational dynamics in the domain of iterated mental paper folding. *Cognitive Systems Research*, *54*, 217–231.

Ashby, F. G., Ell, S. W., Valentin, V. V., & Casale, M. B. (2005). FROST: A distributed neurocomputational model of working memory maintenance. *Journal of Cognitive Neuroscience*, *17*(11), 1728–1743. doi: 10.1162/089892905774589271

Barto, A. G., Singh, S., & Chentanez, N. (2004). Intrinsically motivated learning of hierarchical collections of skills. In *Proceedings of the 3rd international conference on development and learning* (pp. 112–19).

Carruthers, S., Stege, U., & Masson, M. E. (2018). The role of the goal in solving hard computational problems: Do people really optimize? *Journal of Problem Solving*, *11*, 1–19.

Castel, A. D. (2008). Metacognition and learning about primacy and recency effects in free recall: The utilization of intrinsic and extrinsic cues when making judgments of learning. *Memory & Cognition*, *36*(2), 429–437. doi: 10.3758/MC.36.2.429

Choi, E., Kim, C., & Lee, K. C. (2021). Consumer Decision-Making Creativity and Its Relation to ExploitationExploration Activities: Eye-Tracking Approach. *Frontiers in Psychology*, *11*(January), 1–14. doi: 10.3389/fpsyg.2020.557292

Şimşek, Ö. (2020). Bounded rationality for artificial intelligence. *Routledge Handbook of Bounded Rationality*, 338–348. doi: 10.4324/9781315658353-23

Dandurand, F., Bowen, M., & Shultz, T. R. (2004). Learning by imitation, reinforcement and verbal rules in problem solving tasks. In *Proceedings of the third international conference on development and learning: developing social brains* (pp. 88–95).

Davidson, J. E., & Sternberg, R. J. (Eds.). (2003). *The psychology of problem solving*. Cambridge university press.

Drake, J. H., Kheiri, A., Özcan, E., & Burke, E. K. (2020). Recent advances in selection hyper-heuristics. *European Journal of Operational Research*, *285*(2), 405–428.

Dry, M. J., Lee, M. D., Vickers, D., & Hughes, P. (2006). Human performance on visually presented traveling sales–person problems with varying numbers of nodes. *Journal of Problem Solving*, *1*, 20–32.

Fleischer, P., Hélie, S., & Pizlo, Z. (2018). The role of problem representation in producing near–optimal TSP tours. *Journal of Problem Solving*, *11*, 2.

Gardner, J. L. (2019). Optimality and heuristics in perceptual neuroscience. *Nature neuroscience*, *22*(4), 514–523.

Gentner, D. (1983). Structure-mapping: A theoretical framework for analogy. *Cognitive Science*, *7*, 155–170.

Gigerenzer, G. (2000). *Adaptive Thinking: Rationality in the Real World*. Oxford University Press.

Graham, S. M., Joshi, A., & Pizlo, Z. (2000). The traveling salesman problem: a hierarchical model. *Memory & Cognition*, *28*, 1191–1204.

Hélie, S., & Olteteanu, A.-M. (in press). Computational models of creativity. In R. Sun (Ed.), *Cambridge handbook of compu-*

tational cognitive sciences. 2nd edition. Cambridge University press.

Hélie, S., & Sun, R. (2010, jul). Incubation, insight, and creative problem solving: A unified theory and a connectionist model. *Psychological Review*, *117*(3), 994–1024.

Hilbert, D., & Cohn-Vossen, S. (1952). *Geometry and the Imagination*. Providence, RI: AMS Chelsea Publishing.

Jia, X., Li, P., Li, X., Zhang, Y., Cao, W., Cao, L., & Li, W. (2016). The effect of word frequency on judgments of learning: Contributions of beliefs and processing fluency. *Frontiers in Psychology*, *6*(JAN), 1–10. doi: 10.3389/fpsyg.2015.01995

Katsikopoulos, K., Şimşek, Ö., Buckmann, M., & Gigerenzer, G. (2021). *Classification in the Wild: The Science and Art of Transparent Decision Making*. MIT Press.

Korf, R. E. (1985). Depth–first iterative-deepening: An optimal admissible tree search*. *Artificial Intelligence*, *27*(1), 97–109. doi: 10.1016/0004-3702(85)90084-0

Korf, R. E., & Felner, A. (2002). Disjoint pattern database heuristics. *Artificial intelligence*, *134*(1-2), 9–22.

Kornell, N., Rhodes, M. G., Castel, A. D., & Tauber, S. K. (2011). The Ease-of-Processing Heuristic and the Stability Bias. *Psychological Science*, *22*(6), 787–794. doi: 10.1177/0956797611407929

Kunda, M., McGregor, K., & Goel, A. K. (2013). A computational model for solving problems from the Ravens Progressive Matrices intelligence test using iconic visual representations. *Cognitive Systems Research*, *22–23*, 47–66.

Lakoff, G., & Johnson, M. (1999). *Philosophy in the Flesh: The Embodied Mind and its Challenge to Western Thought*. New York: Basic Books.

Lawler, E. L., Lenstra, J. K., Rinnooy Kan, A. H. G., & Shmoys, D. B. (Eds.). (1992). *The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization*. Chichester, UK: Wiley.

Lovett, A., & Forbus, K. (2017). Modeling visual problem solving as analogical reasoning. *Psychological Review*, *124*, 60–90.

MacGregor, J. N., & Cunningham, J. B. (2009). The effects of number and level of restructuring in insight problem solving. *Journal of Problem Solving*, *2*, 130–141.

Miller, T. M., & Geraci, L. (2016). The influence of retrieval practice on metacognition: The contribution of analytic and non-analytic processes. *Consciousness and Cognition*, *42*, 41–50. doi: 10.1016/j.concog.2016.03.010

Mueller, M. L., & Dunlosky, J. (2017). How beliefs can impact judgments of learning: Evaluating analytic processing theory with beliefs about fluency. *Journal of Memory and Language*, *93*, 245–258. doi: 10.1016/j.jml.2016.10.008

Mueller, M. L., Dunlosky, J., Tauber, S. K., & Rhodes, M. G. (2014). The font-size effect on judgments of learning : Does it exemplify fluency effects or reflect people ' s beliefs about memory ? *Journal of Memory and Language*, *70*, 1–12. doi: 10.1016/j.jml.2013.09.007

Newell, A., & Simon, H. A. (1972). *Human Problem Solving*. Prentice–Hall.

Ohlsson, S. (1984). Restructuring revisited: I. Summary and critique of the Gestalt theory of problem solving. *Scandinavian Journal of Psychology*, *25*, 65–78.

Olteteanu, A. M., & Indurkhya, B. (2019). Re–representation in cognitive systems. *Frontiers in Cognitive Science*.

Pizlo, Z., & Li, Z. (2005, sep). Solving combinatorial problems: The 15-puzzle. *Memory & Cognition*, *33*(6), 1069–1084.

Raymond, J. E., Shapiro, K. L., & Arnell, K. M. (1992). Temporary suppression of visual processing in an RSVP task: An attentional blink? *Journal of Experimental Psychology: Human Perception and Performance*, *18*, 849–860.

Rich, P., Blokpoel, M., de Haan, R., & van Rooij, I. (2020). How Intractability Spans the Cognitive and Evolutionary Levels of Explanation. *Topics in Cognitive Science*, *12*(4), 1382–1402. doi: 10.1111/tops.12506

Rouinfar, A., Agra, E., Larson, A. M., Rebello, N. S., & Loschky, L. C. (2014). Linking attentional processes and conceptual problem solving: Visual cues facilitate the automaticity of extracting relevant information from diagrams. *Frontiers in Psychology*, *5*, 1094.

Sajedinia, Z., Pizlo, Z., & Hélie, S. (2019). Investigating the role of the visual system in solving the traveling salesperson problem. In G. Ashok, C. Seifert, & C. Freksa (Eds.), *Proceedings of the 41st annual meeting of the cognitive science society* (pp. 2702–2707). Austin, TX: Cognitive Science Society.

Schooler, J. W., Ohlsson, S., & Brooks, K. (1993). Thoughts beyond words: When language overshadows insight. *Journal of Experimental Psychology: General*, *122*(2), 166–183. doi: 10.1037/0096-3445.122.2.166

Searle, J. R. (1980). Minds, brains, and programs. *Behavioral and Brain Sciences*, *3*, 417–457.

Serra, M. J., & Ariel, R. (2014). People use the memory for past-test heuristic as an explicit cue for judgments of learning. *Memory & Cognition*, 1260–1272. doi: 10.3758/s13421-014-0431-0

Shah, A. K., & Oppenheimer, D. M. (2008). Heuristics made easy: An effort-reduction framework. *Psychological Bulletin*, *134*, 207–222.

Simon, H. A. (1972). Theories of bounded rationality. In C. B. McGuire & R. Radner (Eds.), *Decisions and organizations* (pp. 161–176). North–Holland Publishing Company.

Smith, S. F. (1983). Flexible learning of problem solving heuristics through adaptive search. In *Ijcai* (Vol. 83, pp. 422–425).

Sutton, R., & Barto, A. (1998). *Reinforcement Learning: An Introduction*. MIT Press.

Wang, L., Zhang, D., Zhang, J., Xu, X., Gao, L., Dai, B. T., & Shen, H. T. (2019). Template-based math word problem solvers with recursive neural networks. In *Proceedings of the aaai conference on artificial intelligence* (Vol. 33, pp. 7144–7151).

Yahosseini, K., & Moussaïd, M. (2019). Search as a simple take-the-best heuristic. *Royal Society open science*, *6*(10), 190529.