The Role of the Goal in Solving Hard Computational Problems: Do People Really Optimize?

Sarah Carruthers, Ulrike Stege, & Michael E. J. Masson

2018

SPECIAL ISSUE

Journal of Problem Solving

The Role of the Goal in Solving Hard Computational Problems: Do People Really Optimize?

Sarah Carruthers,[1] Ulrike Stege,[1] and Michael E. J. Masson[2]

[1] Department of Computer Science, University of Victoria, Victoria, BC, Canada
[2] Department of Psychology, University of Victoria, Victoria, BC, Canada

Correspondence:
Correspondence concerning this article should be addressed to Sarah Carruthers, University of Victoria, via email to scarruth@uvic.ca

Keywords:
problem solving, computational complexity, intractability

The role that the mental, or *internal*, representation plays when people are solving hard computational problems has largely been overlooked to date, despite the reality that this internal representation drives problem solving. In this work we investigate how performance on versions of two hard computational problems differs based on what internal representations can be generated. Our findings suggest that problem solving performance depends not only on the objective difficulty of the problem, and of course the particular problem instance at hand, but also on how feasible it is to encode the goal of the given problem. A further implication of these findings is that previous human performance studies using NP-hard problems may have, surprisingly, underestimated human performance on instances of problems of this class. We suggest ways to meaningfully frame human performance results on instances of computationally hard problems in terms of these problems' computational complexity, and present a novel framework for interpreting results on problems of this type. The framework takes into account the limitations of the human cognitive system, in particular as it applies to the generation of internal representations of problems of this class.

INTRODUCTION

The human cognitive system is capable of dealing with complex, including objectively difficult, problems often with seemingly little effort. Everyday tasks such as scheduling workers or resources, and packing groceries efficiently are closely related to problems that are hard in an objective sense. Scheduling tasks to minimize conflicts, for example, can be equivalent to the NP-complete vertex cover problem; packing the most items in a container is akin to the NP-complete bin packing problem (Arora & Barak, 2007; Garey & Johnson, 1979). The human cognitive system's ability to tackle complex problems like these with what can seem like little or no effort is fascinating and deserving of continued study.

One of the original motivations of this study was to evaluate human performance on computational problems that, unlike the previously studied Euclidean traveling salesperson problem, were *not Euclidean* in nature.

When solving an instance of the Euclidean traveling salesperson problem one is presented with a point set in the Euclidean plane. The task is to find a shortest tour visiting all points, where the tour length is measured using the Euclidean distance. The problem can also be viewed as a Euclidean graph problem: a Euclidean graph is one where the vertices are points in Euclidean space. The edge length is given by the Euclidean length between their end vertices. Alternatively, for graphs that are not Euclidean and therefore are not embedded in the Euclidean space, the distance between points is not relevant to the problem definition. Such a graph that is not embedded in space has infinitely many different configurations, which are all equivalent.

There are a number of different explanations given in the literature to explain performance on the well-known Euclidean traveling salesperson problem. These explanations can be divided into two categories, either perceptual or analytical (as proposed by van Rooij, Schactman, Kadlec, & Stege, 2006).

Studying performance on hard computational problems that are not Euclidean provides another mechanism to evaluate whether perceptual processes that depend on the Euclidean nature of the problem can alone explain performance. If similar human performance results are observed on these problems, this would support the hypothesis that the reported high quality and fast timing results were not merely a result of visual problem solving processes. This alone does not discount the possibility that the visual system is not somehow being leveraged (as proposed by Pizlo et al., 2006) but rather would serve to discount certain mechanisms as the sole source of solution quality as these mechanisms do not directly apply to instances of graph problems that are not Euclidean.

Another purpose of this study was to evaluate the role that the specific goal of a problem plays in influencing

http://dx.doi.org/10.7771/1932-6246.1200

performance on hard computational problems by tasking participants with hard computational problems that differed only in terms of how the goal was described. Here, we use the term goal in the formal sense proposed by Newell and Simon (1972), where problem solvers navigate a problem space in search of the goal. Participants were tasked with instances of versions of the vertex cover or independent set problem with stated goals in different manners. These two problems, like the Euclidean traveling salesperson problem, are graph problems that (for the formulations used in this work) are known to be hard to solve, in general (Garey & Johnson, 1979). Their formal definitions are given in Section 1.3.

For NP-hard problems, no algorithms have been found to date that would determine solutions to instances of any finite size in a polynomial amount of time (here the polynomial has as parameter the size of the instance).

PREVIOUS WORK

One approach to investigating how the human cognitive system is able to cope with complexity is to study human performance on instances of computational problems that have been shown to be objectively hard. One such measure of objective difficulty is found in the study of computational complexity; problems that are NP-hard[1] or NP-complete[2] (Garey & Johnson, 1979).

Work to date has found that people are able to find good solutions to instances of classically difficult problems quite quickly. One such problem is the Euclidean traveling salesperson problem (E-TSP) which asks for a shortest tour that visits every given city (or point in the plane) exactly once, starting and finishing at the same location. Studies of human performance on instances of the optimization version of E-TSP have found that humans typically find solutions that are close to optimal (Best & Simon, 2003; Dry, Lee, Vickers, & Hughes, 2006; Dry, Preiss, & Wagemans, 2012; Graham, Joshi, & Pizlo, 2000; Kong & Schunn, 2007; Ormerod & Chronicle, 1999; Saalweachter & Pizlo, 2008), taking close to linear time (in terms of the number of points in the input) to complete (Dry et al., 2006, 2012; Graham et al., 2000). A study of the minimum vertex cover problem (a computationally hard problem that is not Euclidean in nature) (Carruthers, Masson, & Stege, 2012) focused on properties of instances that impact human performance, and what strategies participants might adopt when tackling instances. Performance on instances of this problem was in line with performance results reported on the E-TSP, ranging, depending on the factor, from roughly 4% to 10% above optimal.

Many of these findings are seemingly at odds with the current understanding of the complexity of this problem: problems like E-TSP and minimum vertex cover are hard, and therefore no fast algorithm[3] likely exists that can solve arbitrary instances of them. There are even strict bounds on how close approximation algorithms[4] can get to the optimal path of E-TSP (Karpinski, Lampis, & Schmied, 2013), and many approximation algorithms require more than the linear time observed in human performance studies.

Explanations for these findings vary. In the case of E-TSP, it has been proposed that Gestalt principles of continuation and good form (Frisby & Stone, 2010) may be, at least partially, responsible for the solutions that participants find (Tak, Plaisier, & van Rooij, 2008). A hierarchical pyramid model that uses clustering produces solutions that closely match those found by participants (Graham et al., 2000). Whatever the explanation, we observe that all studies to date using hard computational problems as instruments to study human problem solving performance have been limited to their *optimization* version: problems for which the aim of the task is to find a *best* solution. In the case of E-TSP, for example, problem solvers are tasked with finding a *shortest* route that visits all vertices in a graph. One of the aims of this work was to investigate human performance on hard computational problems, which would not be as vulnerable to Gestalt principles of continuation and good form as E-TSP.

INFORMATION-PROCESSING MODEL OF PROBLEM SOLVING

Let us step back in time to revisit foundational problem-solving research that, as will be shown, is still highly relevant today. The advent of powerful computers in the 1950s led to the rise in popularity of computational theories of mind, and is a great example of how the emergence of new tools can influence the advance of scientific theories (Gigerenzer & Goldstein, 1996). It is easy to take for granted the concept of thinking about problem solving as a computational task, but at the time being able to think about problem solving from a computational perspective allowed for a wealth of knowledge about problems and problem solving from the field of computer science to be transferred to the study of human problem solving. Newell and Simon's (1972) information-processing model of human cognition, and in particular problem solving, motivated a great number of experimental studies using problems of transformation. These problems of transformation are, importantly, problems that are *well defined.* A well-defined problem has start and goal states, and a finite set of legal operators (Newell & Simon, 1972). Further, these states and operators must be *encodable* in the problem solver's internal representation. Examples of well-known problems of transformation include: towers of Hanoi, water-jug problems, and chess. One of the aims of this work is to revisit and, if necessary, revise these definitions in the context of hard computational problems. This will be addressed in Section 1.5.

One of the fundamental concepts that came out of the information-processing model of problem solving is that of a *problem space.* When a problem solver is tasked with a problem, they generate an internal representation, or

problem space (Newell & Simon, 1972), that can be thought of as a representation of all possible paths from the start state, through all reachable states, to all goal states.[5]

So long as the given legal operators are sufficient to ensure that there exists at least one path from the start to the goal, then it can be assumed that the problem solver's problem space also contains a path from the start to the goal, and therefore the problem is solvable.

At this juncture it is also important to note the role of backtracking in finding solutions to problems of this kind. Finding a solution to instances of hard problems is rarely done by choosing a series of only correct moves, that is, moves that are all on a path yielding an optimum solution. For non-trivial instances it is natural that problem solvers make moves that do not directly lead to the solution, resulting in *backtracking* moves. The practice is largely overlooked in the literature. It is implicitly described in Newell and Simon's (1972) description of search space, which includes all moves, both correct and incorrect. Backtracking can reflect a problem solver exploring the problem space, that is, trying moves to gain an understanding the consequences of a move. Backtracking can also be an indication of a purposeful strategy, or of a problem solver recognizing when they are no longer on a path to the goal.

In either case, measuring the number of moves including both forwards and backwards moves can provide important insight into a problem solver's strategy. Note that backtracking is also part of algorithm-design strategies for exact algorithms solving NP-hard problems. One of these approaches is the technique of bounded search trees in the area of parameterized complexity (Downey & Fellows, 2012).

ALTERNATIVES TO E-TSP: VERTEX COVER AND INDEPENDENT SET PROBLEMS

One of the goals of this work was to investigate human performance on hard computational problems that would not be as vulnerable to Gestalt principles of continuation and good form as E-TSP. Two problems that are not Euclidean were selected for this study: the vertex cover problem and the independent set problem. Both of these problems, like E-TSP, can be visually presented as graph problems, and are also both hard.

The vertex cover problem asks, given a graph, for a smallest set of vertices such that every edge in the graph is incident to at least one of these vertices. Figure 1 shows an optimal vertex cover. More formally, we can define the vertex cover problem as follows:

Vertex Cover (optimization version)

Input: A graph $G = (E, V)$, with vertex set V, and an edge set E.
Output: A vertex cover for G where the size of the vertex cover is minimized. That is, a subset U of V, such that every edge in E is incident to at least one vertex in U, where the size of U is minimized.

The independent set problem asks for a maximum independent set, a biggest set of vertices that are not connected by any edges.[6]

Independent Set (optimization version)

Input: A graph $G = (E, V)$, with vertex set V, and an edge set E.
Output: An independent set for G where the size of the independent set is maximized. That is, a subset W of V, such that no two vertices in W are adjacent, where the size of W is maximized.

ALTERNATIVES TO OPTIMIZATION: SEARCH AND DECISION PROBLEMS

One observation is that human performance studies on hard computational problems found in the literature have focused exclusively on optimization problems. The optimization version of a problem asks for a *best* solution, either minimizing or maximizing some measure. For instance, the E-TSP asks for a *shortest tour*. Optimization has long been a topic of interest for cognitive scientists, and it has been suggested that some types of optimization problems may exceed the limits of the human cognitive system (Simon, 1990). An optimization strategy can require the problem solver to store and compare a great number of options in order to determine a solution that is definitively optimal. Given problems that demand too much of the cognitive system, Simon's theory of bounded rationality suggests that people turn to either satisfying heuristics or fast and frugal heuristics (Todd & Gigerenzer, 2000). The former, like optimization, works by comparing alternatives until one that is satisfactory is found, the latter using little information or computation. The former may apply to hard computational problems like those of interest here.

However, before jumping to the conclusion that when faced with hard optimization, problem solvers resort to using heuristics (van Rooij & Wareham, 2012; van Rooij, Wright, & Wareham, 2012), observe that if optimization exceeds the limits of the cognitive system, then it may be appropriate to study human performance on hard problems that do not include optimization as part of the goal. Fortunately, there are other formulations of computational problems that offer just that possibility. Complexity classes like NP-hard and NP-complete, for instance, are based on the *decision version* of problems, not the optimization versions. In this study, in addition to the commonly used optimization version of the problem, an alternative problem formulation, the search version, was also used. The following sections introduce two problem formulations commonly used in the field of computer science, the decision and search versions.

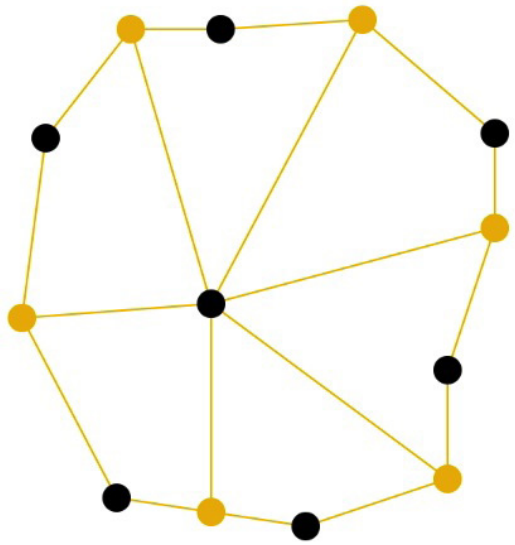

Figure 1
An optimal vertex cover of size 6 to a graph with 13 vertices and 18 edges. Yellow vertices are in the cover. Yellow edges are covered by those vertices.

Decision Version

The decision version of a computational problem asks for a YES/NO answer to the existence of a solution to the problem. Consider the vertex cover problem defined earlier, that asked for a smallest set of vertices. The decision version of the vertex cover problem asks if there exists a valid vertex cover with at most a specified number of vertices.

Vertex Cover (decision version)

Input: A graph $G = (E, V)$, with vertex set V, edge set E, and a positive integer k.
Output: The *YES*/*NO* answer to the question: Does there exist a vertex cover of size at most k?

The optimization version of the independent set problem asks for a largest set of vertices. The decision version on the other hand asks about the existence of an independent set, of at least a given size.

Independent Set (decision version)

Input: A graph $G = (E, V)$, with vertex set V, and edge set E. A positive integer value k.
Output: The *YES*/*NO* answer to: Does there exist an independent set for G, of size at least k?

Search Version

In addition to the optimization and decision versions of problems, there is a third version of problem commonly used in computational complexity: the search version. It asks about the existence of a solution given some criteria (like the decision version), and if one exists it asks for a solution (like the optimization version). The search version is in many ways a better problem variant of the decision problem because knowing a solution is way more informative than just knowing that there is one. The search version of vertex cover, for instance, asks for a cover of size at most k, if one exists, or the answer NO, if not.

Vertex Cover (search version)

Input: A graph $G = (E, V)$, with vertex set V, edge set E, and positive integer value k.
Output: A vertex cover for G of size at most k. Otherwise output "no solution exists."

Independent Set (search version)

Input: A graph $G = (E, V)$, with vertex set V, and edge set E. A positive integer value k.
Output: An independent set for G, of size at least k, if one exists, otherwise output "no solution exists."

Complexity of the Optimization and Search Versions of Computational Problems

Computational complexity results are typically based on the decision version of problems; however it is not uncommon to see claims that the optimization versions of E-TSP (and variants) are *NP-complete* or *NP-hard*. This is not strictly true; however, the optimization or search version of many problems can be converted (in polynomial time in terms of the input size) to the decision version, in which case the complexity of the search or optimization versions can be thought of as equivalent to that of the decision version. For example, if the optimization version of vertex cover is easy, then it could be used to solve the decision version of vertex cover in polynomial time, which would contradict the result that the decision version of vertex cover is *NP-complete*.

We have now introduced two hard computational problems that appear less susceptible to visual processes due to their not being Euclidean in nature. Computational problems like the vertex cover problem and the independent set problem are free of unknowns and uncertainty and are therefore good candidates as instruments to better understand problem solving, because it is possible to understand the problem very well at the computational level. This, of course, depends upon the assumption that the computational problem given is in fact the problem that is being *solved* by the problem solver. We now identify why this might not be the case for the problems that have been the focus of study to date.

CAN HARD OPTIMIZATION PROBLEMS BE WELL DEFINED?

Computational problems, like the vertex cover and independent set problems, are well defined at least as far as computer scientists are concerned, and this would seem to imply that they must also be well defined as far as the human problem

solving definition given above is concerned. However, in order to be well defined in the latter sense, the goal must be verifiable; that is, the problem solver must be able to encode the goal state in order to determine if and when it has been achieved. This, as will be shown, may not in fact be the case for instances of many hard optimization problems.

Consider the optimization version of the vertex cover problem. Determining that a given vertex cover for a particular graph is optimal requires, given a candidate solution[7] of size *k*, determining with certainty that there does not exist a valid solution of smaller size. If even only a small fraction of other candidate covers are considered to answer this question, this is well beyond the limits of working memory for all but trivial instances of the problem. For non-trivial instances, even comparing only a small number of covers quickly exceeds the limits of working memory (Baddeley & Hitch, 1974). It is possible that there exist other means of determining that a candidate cover is optimal for given instances, but it is not immediately clear what they may be. Again, since participants often fail to find optimal solutions, it is unlikely that such an alternative is readily available to novice problem solvers. Therefore, it is assumed that in general the task of determining the optimality of a candidate vertex cover is infeasible, and a goal state for this problem is ill-defined. Similarly, determining that a valid independent set is optimal amounts to showing that no larger independent set exists, a task that is infeasible, as the goal state for this problem definition is ill-defined.

It turns out that some aspects of a goal state of hard optimization problems are infeasible to evaluate, in general (Carruthers, 2015). This is worth noting as this likely also holds for the problem of chess used by Newell and Simon (1972). Despite this, problem solvers, when presented with an optimization problem, generate an internal representation of the problem that allows them to determine when a goal state has been reached. It is therefore key to identify what goal(s) might be encoded within the internal representation, as this defines what problem is being solved. This in turn determines a sound base from which algorithmic explanations of human performance can be proposed.

Consider, again, the problem solver's internal representation of the problem. It consists of a problem space, defined by the start state, legal operators, and goal state of the problem. For many of the hard optimization problems studied to date, the start state and legal operators are well defined, and therefore is it possible that they are consistent with those in the internal representation. Identification of a goal state, on the other hand, has been shown to be infeasible and cannot be encoded as such in the problem space. In studies of human performance on hard optimization problems, participants identify solutions to most (if not all) instances presented. These solutions, often suboptimal, are selected because they satisfy some requirement or quality. Some solutions may also be the result of the problem solver giving up on the task of finding a state that matches a goal state, but this does not exclude the existence of a feasibly identifiable goal state. Suboptimal performance results on hard optimization problems have typically been explained by proposing that problem solvers use approximations or heuristics (Dry et al., 2006, 2012; Graham et al., 2000; MacGregor, Chronicle, & Ormerod, 2004, 2008; MacGregor, Ormerod, & Chronicle, 2000; Pizlo et al., 2006). However, this approach ignores the underlying issue of what problem is being solved; it ignores how the problem represented internally determines when a solution matches the goal. According to the Newell and Simons (1972) information processing account, the problem solver determines that the instance of the problem is solved when a goal state is achieved, which is not possible if it is assumed that the original (infeasible-to-evaluate) goal state is encoded in the internal representation.

Many different kinds of problem exist that might be suitable for the study of human problem solving. Problems of transformation, free of uncertainty and unknowns, have proven useful for investigating how problem solvers navigate through problem spaces guaranteed to contain a path to the goal. In other words, they have made it possible to study performance on problems that are subjectively difficult, but solvable. Work in this area on problems whose problem spaces are relatively small has yielded interesting insights about factors that contribute to the subjective difficulty of these problems. It is also important to study how people cope with problems that, while still solvable, are more complex, problems that are objectively difficult. Hard optimization problems seemed to offer such a possibility, as they are very closely related to computational problems for which no efficient algorithm exists. Rather surprisingly, human performance on instances of these objectively difficult tasks was found to be very good, with problem solvers finding near-optimal solutions very quickly, in many cases. However, meaningful interpretation of these performance results hinges on the assumption that problem solvers are indeed solving the problem given. If problem solvers are not able to encode part of these hard optimization tasks in their internal representation, then these performance results may not in fact be indicative of performance on the given task, but rather on some other, unknown, task.

A REVISED APPROACH TO STUDYING HUMAN PROBLEM SOLVING OF HARD PROBLEMS

Finding a solution to most instances of the optimization version of some hard computational problems is likely beyond the power of the human cognitive system because the ill-defined goal is not encodable. If the given problem cannot be encoded then the problem solver cannot be

solving the assigned task. While this does not exclude the existence of solvable instances of any non-encodable problem, the computational challenge of identifying the goal state renders countless other instances infeasible to solve. The seeming contradiction of using non-encodable problems as instruments to evaluate human problem solving does not preclude the possibility of continuing to use hard computational problems to gain an understanding of the power of the human cognitive system; however, it does imply that it may be appropriate to either modify the way these kinds of problem are presented or modify how performance results are evaluated. Two possible improvements to the existing problem-solving study design are identified. The first is to continue using hard optimization problems as instruments and, based on the observation that these problems are not encodable, attempt to identify what problem(s) might be encoded before analyzing performance results. The second is to identify hard computational problems that are encodable by the human cognitive system and use them to evaluate human performance on hard computational problems. There are benefits to both approaches.

Hard problems, in particular those for which it may not be possible for the problem solver to determine if a solution is correct, are everyday occurrences of the environment in which the human cognitive system has evolved. As a result, it would be surprising if the human cognitive system had not developed fast and efficient ways to cope with hard problem-solving situations. Using problems that are not necessarily encodable by the human cognitive system can allow for the investigation of possible mechanisms that the cognitive system can use to render the problems encodable.

On the other hand, hard problems that are theoretically encodable by the cognitive system, and are difficult due to their computational complexity alone, can be used as tools to investigate how problem solvers navigate problem spaces in which they are unable to easily determine a path to the goal. For even very hard problems, there can exist rules or strategies which can be used to reduce the size of the problem space of some instances. Encodable hard problems could allow us to observe whether or not problem solvers are able to identify, learn, and apply these strategies on later instances.

Previous work on problems of transformation proposed a number of general problem-solving strategies, like hill climbing, brute force, and means–ends analysis, as possible explanations of human performance on problems of transformation. While identifying such general problem-solving strategies is important, a different approach is considered here. This is in part due to the fact that it is unlikely that a general problem-solving strategy can be sufficient to find solutions to instances of problems that are shown to be computationally hard. Instead, people may acquire and apply strategies specific to classes of instances, and possibly acquire associated schemata.

A Framework for Handling the Ill-Defined Goal of Hard Optimization Problems

Human solutions to instances of hard optimization problems can be close to optimal and even optimal. It may therefore seem valid to assume that participants do find solutions to the given task, as participants typically demonstrate understanding of the task and do not verbalize that they are trying to solve a problem other than the one given. This, however, is likely a reflection of the adaptivity of our cognitive system. On the surface, this seems to be what Gigerenzer (2001) is referring to with *fast and frugal heuristics*: if a problem cannot be solved exactly then heuristics are a natural alternative. However, before assuming that heuristics or some other suboptimal strategies are the only way to explain performance, it is important to determine what problem might be encoded by the problem solver because the encoded problem drives performance.

When tasked with instances of hard optimization problems that are not encodable by the cognitive system, the problem solver encodes *some* problem, as indicated by their ability to find a solution. These solutions can be very close to optimal, which implies that the encoded problem likely shares a great deal with the original unencodable problem. Below we propose a way to identify alternative problem formulations which are encodable by the human cognitive system. Identifying candidate problems that can be encoded by the cognitive system makes it possible to propose models that might predict performance. This process begins with decomposing the given problem's goal into those aspects that are feasibly identifiable and those that are not.

For the minimum vertex cover problem, the goal is to find a vertex cover of minimum size. We can decompose this goal into two components, one that is feasible to identify, and another that, for most instances, is infeasible to evaluate. It is feasible to determine whether a given candidate vertex cover is valid: one can consider each edge in the graph independently, and check whether it is incident to at least one vertex in the candidate cover. Each "edge check" in this process is an independent one, and the load on working memory for its execution is low. Therefore, determining whether a set of vertices is a vertex cover is feasible. Determining whether a candidate cover is smallest, however, is in general not feasible, as discussed earlier.

From this decomposition, the aspect of the goal that is infeasible to evaluate can be clearly identified and potentially modified into one that is feasible to identify. It is important to note that most modifications of the goal for a problem will result in a different problem. For any problem, there may be many possible ways such a modification can take place. An idealist approach is to assume that the goal is modified

as parsimonious as possible while retaining as much of the original goal structure as possible. This is in line with the idea that only infeasible-to-evaluate aspects are modified. It is possible that goal modifications do not adhere to this idealist perspective; however, it is a prudent starting point. Regardless of the level of modification, three distinct categories of goal modification are identified: restructuring modifications, general modifications, and specific modifications. Two problems, vertex cover and independent set, are analyzed in terms of these goal modifications.

Goal Modification Types

Restructuring Modifications. It may be possible for goal aspects, both feasible and infeasible, to be recombined by the problem solver in such a way that the goal is rendered feasible to identify. Local optimization is an example of such a restructuring mechanism, where the global optimization requirement is reduced to a local optimization, by shifting the target of optimization. This can be accomplished by recombining goal aspects in such a way that it is feasible to evaluate the new goal given the constraints of the cognitive system.

General Modification. A general modification is one that replaces an aspect of the goal with a general quality that can be feasibly evaluated, without including an exact value or measure. In this case, the infeasible aspect of the goal is either eliminated or replaced with one that can be feasibly evaluated.

Specific Goal Modification. A specific goal modification is one in which an unidentifiable aspect of the goal is replaced with one or more quantitative measures or values. Such values can be refined and/or adjusted depending on whether or not a solution that satisfies the (previously) selected goal value can be found, resulting in an iterative strategy to solving the problem. The problem solver might estimate a value and attempt to find a solution that satisfies that value. If successful, the problem may either be deemed solved or a closer-to-optimal value may be chosen and the process might continue. If it fails, then the problem solver may choose a further-from-optimal value, and the process can continue. This modification results in an internal representation that is very close to the original optimization problem, except that the problem solver explicitly encodes the process of optimization by iteratively attempting to find solutions that are progressively closer to optimal. This would likely manifest in a goal searching mechanism that would involve a great deal of backtracking as a result of searching for a solution that matches the initial (and possibly subsequent) goal.

Goal Modification in Vertex Cover and Independent Set Problems

We now evaluate possible general, restructuring, and specific goal modifications, by decomposing the goal requirements for both vertex cover and independent set problems; candidate modifications are presented based on these decompositions; and predictions for performance based on these modifications. For an overview of predicted performance for each suggested goal modification, see Tables 1 and 2.

Minimum Vertex Cover. As illustrated above, the infeasible aspect of the goal in the minimum vertex cover problem is not that of finding a vertex cover (a vertex cover can be easily found by simply picking one vertex for each edge), but in determining the optimality of the vertex cover.

Restructuring Modifications. The goal can be restructured such that local minimum vertex covers are found in sub-regions of the graph, manifesting as a kind of local optimization. If the subgraphs are small enough, then exhaustive search for a vertex cover becomes feasible, rendering the goal feasible to encode. This restructuring, while feasible, is challenging, as there may not be clearly identifiable sub-regions (or subgraphs) for a given graph. Further, sub-regions may not be mathematically well defined by participants. Some instances may contain dense clusters (particular sub-regions) that are only loosely connected to each other. These may denote, visually, clear boundaries of subgraphs. On the other hand, graphs with relatively even density of edges may be more difficult to consistently decompose into sub-regions. As a result, in general, it may be difficult to maintain a mental model of how an instance is divided into sub-regions. A second consequence is that this division into subgraphs may not be static,

Table 1

Predicted performance for suggested goal modifications for the minimum vertex cover problem. Predictions for each candidate modification are: the solution quality (relative to optimal), the number of optimal solutions found, and the amount of backtracking needed to find a solution.

Modification type	Modification	Solution quality	Opt. solutions	Backtracking
General	Minimal VC	Sub-optimal, minimal	Few	Little to none
General	Any VC	Sub-opt., poss. non-minimal	Few	None
Restructuring	Local optimization	Sub-opt., minimal	Few	Moderate amount
Specific	Guess and check	Near-optimal	Many	Greater amount

Table 2
Predicted performance for suggested goal modifications for maximum independent set problem. Predictions for each candidate modification are: the solution quality (relative to optimal), the number of optimal solutions found, and the amount of backtracking needed to find a solution.

Modification type	Modification	Solution quality	Opt. solutions	Backtracking
General	Maximal IS	Sub-opt., maximal	Few	Little to none
General	Any IS	Sub-opt., poss. non-maximal	Few	None
Restructuring	Local optimization	Sub-opt., maximal	Few	Moderate amount
Specific	Guess and check	Near-optimal	Many	Greater amount

changing during the problem-solving process. Identifying whether and when new subgraphs might be selected could be difficult. This modification predicts sub-optimal vertex covers, as locally optimal choices (in terms of subgraphs) may negatively impact the optimality of the overall solution. It also predicts some backtracking, as the problem solver attempts to find locally optimal covers for each subgraph or adjusts a minimum cover in one subgraph due to the impact of choices in a different subgraph.

General Modification. A general modification to the infeasible aspect of the goal is to find a *minimal* vertex cover. A minimal vertex cover is a valid vertex cover that contains no redundancy, that is, no single vertex can be removed from the cover without violating the requirements of a vertex cover. All optimal vertex covers are minimal. However, not all minimal vertex covers are optimal. For instance, in Figure 2, the vertex cover comprising the colored vertices is minimal, but not optimal. This property can be evaluated by examining each vertex in the cover individually, and determining whether its removal would render the cover invalid.

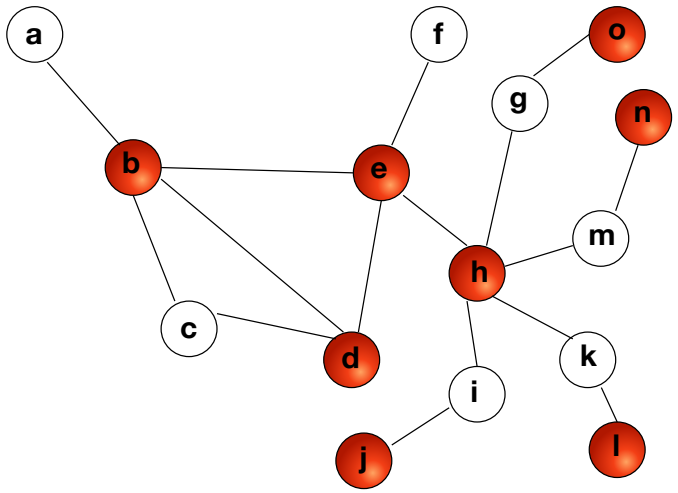

Figure 2
A minimal, and non-optimal, vertex cover. Vertices colored red constitute a valid vertex cover.

The memory requirements for this evaluation are small, as each choice is local and it is dependent only on the vertices adjacent to, and edges incident to, that vertex. This modification would predict that upon finding a minimal cover, participants would not attempt to improve upon this solution, having matched this goal. This modification predicts participants finding fewer minimum vertex covers than if they are working on the given optimization task. It also predicts little or no backtracking once a minimal cover is found. To see why this is the case, consider the case where the problem solver searches for a minimal cover, which can be achieved by making local choices, for instance, finding a vertex that is not yet included in the cover that covers an uncovered edge. This strategy alone is sufficient for finding a minimal cover and requires no backtracking. Finding an optimal cover, however, may require trying to improve upon a minimal cover, which necessitates backtracking and trying different options from some earlier choice. An alternative general modification scheme involves discarding the optimization aspect of the goal entirely and simply searching for any vertex cover. This modification predicts suboptimal solutions, with little or no backtracking. However, it is not as desirable a modification as the previous one, as it less parsimoniously modifies the aspects of the goal by discarding any sense of optimization completely.

Specific Goal Modification. Problem solvers may alternatively resort to a guess and check method to generate a feasibly measurable goal, and as a result they might generate a specific goal modification. This method seems especially appealing in problems with integer value measures, like vertex cover or independent set. Given an instance, a specific value k could be chosen by the participant, who could then search for a vertex cover no larger than k. If one is found, the solution might be deemed satisfactory or the value could be modified and another (better) solution could be searched for. Similarly if a solution is not found, the problem solver could either give up, or choose a further-from-optimal value and reiterate the process. This modification predicts that the amount of backtracking will be related to the optimality of the solutions. That is, more backtracking will be required if close-to-optimal specific

values are chosen, and less backtracking will be required to find solutions that are not close to optimal. This modification results in a process of iteratively solving the search version of the problem for different values of k.

Maximum Independent Set. Like vertex cover, the infeasible aspect of the goal in the maximum independent set problem is that of optimizing the size of the independent set. Due to the similarities these two problems share, the modifications are likewise similar.

Restructuring Modifications. The infeasible-to-evaluate goal could be restructured, resulting in searching for a maximal independent set. A maximal independent set is a valid independent set which cannot be made larger by simply adding another vertex to the set. This modification predicts that problem solvers would terminate search upon finding a maximal independent set, with little or no backtracking. An alternative general modification scheme is to find any independent set, such that the optimization goal aspect is discarded entirely. Due to the nature of the independent set problem, an empty set is an independent set, and therefore this modification is undesirable. It predicts potentially greatly suboptimal and even non-maximal solutions, with little or no backtracking.

General Modification. The local optimization maximum independent set problem, where maximum independent sets are identified in subgraphs of the given instance, is also a viable candidate restructuring. It is vulnerable to the same issues as the local minimum vertex cover problem described above. It predicts suboptimal solutions, and even non-maximal solutions, with some backtracking.

Specific Goal Modification. Like minimum vertex cover, a specific goal modification is possible for this problem. Given an instance, a specific value k could be chosen and the problem solver could then search for an independent set with at least k vertices. If one is found, the solution might be deemed satisfactory or the value could be modified and another (better) solution could be searched for. If an independent set of size k is not found, the problem solver could either give up or choose a further-from-optimal value and reiterate the process. Predictions for this guess-and-check modification are similar to that for vertex cover.

Candidate modifications have been identified for the optimization versions of the vertex cover and independent set problems. These proposed modifications, unlike the original optimization version of the problems, are encodable by the cognitive system. In both cases, the restructuring (minimal/maximal) and specific modifications are the most appealing because they maintain more of the original problem structure than the general modification, and also predict close-to-optimal solutions. Both these modifications predict close-to-optimal solutions, but differ in the amount of backtracking they predict. These predictions provide quantifiable measures for comparison to human performance results or computational model results.

METHODOLOGY

This study and the methodology chosen were based in part on pilot studies of human performance on the optimization version of both the vertex cover problem and the independent set problem, as described earlier. Participants were presented with instances of the given problems on an iPad which made it possible to provide cognitive support to the participants not previously provided in the pilot studies.

PARTICIPANTS

Ninety-six undergraduate students participated in the study at the University of Victoria for optional extra credit in a psychology course. Ethics approval for this study was obtained through the University of Victoria, and all participants gave informed consent before participating.

MATERIALS

A set of 25 graphs was presented to all participants. The instrument included 10 small graphs with 10–19 vertices each, 10 medium graphs with 20–29 vertices, and 5 large graphs with 30–45 vertices. Graphs were generated according to predetermined properties as described by Carruthers (2015).

PROCEDURE

Participants were randomly assigned to conditions as follows. Half of the participants were instructed to solve the vertex cover problem and the other half were instructed to solve the independent set problem. Within each of these conditions half of participants were assigned to the Optimization (OPT) condition and half were assigned to the Search (SRC) condition. See Table 3 for details.

The vertex cover problem was presented as the guard problem. Participants were told that the graphs represented art galleries where the vertices were guard posts and the edges were the hallways containing priceless art. Participants in the Optimization group were asked to find the fewest guards

Table 3
Number of participants randomly assigned to each condition.

	Vertex cover	Independent set	Total
Optimization version	24	24	48
Search version	24	24	48
Total	48	48	96

needed to cover all the hallways. Participants in the Search group of the vertex cover problem were asked whether they could find a way to cover all the hallways with at most a given number of guards. If they deemed the task not possible they entered "NO" into the "goal-possible" field. For all graphs the value given was the size of a minimum vertex cover so that participants in both vertex cover conditions were tasked instances of problems for which identical solutions were valid.

The independent set problem was presented as the independent people problem. Participants were told that the graphs represented social networks and that the vertices were people and the edges were relationships between people, that is, whenever two people are connected by an edge, they know each other. Participants in the Optimization group were asked to find the largest group of people who did not know each other. Participants in the Search group of the independent set problem were asked whether they could find a group of at least a given number of people who did not know each other. If they deemed the task was not possible they were instructed to enter "NO" into the "goal possible" field. In all instances the value given was the size of a maximum independent set so that all participants in both independent set problem conditions were tasked with the same goal.

Factors

The analysis presented in this work considers a number of factors and their interactions, and a number of measures. The factors considered in this paper are:

- Problem (independent set versus vertex cover)
- Problem Version (Search versus Optimization)

The Problem factor compares performance between two different problems given the same graphs. The Problem Version factor investigates differences in performance between the Search and Optimization versions, as well as proposed goal modifications. The interaction between Problem and Problem Version factors investigates how differences in performance between the Search and Optimization versions manifest between these two different problems.

Performance Measures

A number of performance measures are considered, including:

- Percent Optimal solutions
- Percent Above Optimal (PAO)
- Percent Maximal/Minimal solutions
- Number of Moves
- Time per Move
- Number of Undos
- Number of Toggles

Solution quality was investigated using a number of different measures. In all conditions, the percent of optimal solutions was calculated. This value is appropriate for comparing performance between all conditions. In the Optimization condition of both problems, PAO was computed to compare the relative performance between problems, and also to compare performance on these not-Euclidean problems to previously studied hard computational problems. Calculation of PAO is described below. In addition, the percent of maximal (independent set problem) and minimal (vertex cover problem) solutions found was computed. While these measures are not necessarily an indication of correct or perfect solutions to the problems given, they are a relative measure of performance.

It is common to use standardized or normalized measures when comparing performance on hard computational models between conditions (Graham et al., 2000; MacGregor & Ormerod, 1996; van Rooij et al., 2006; Vickers, Butavicius, Lee, & Medvedev, 2001). In this work PAO is computed for the Optimization condition of both problems. For the vertex cover problem, the PAO is computed as:

$$S_{VC} = \frac{S_{obs} - S_{opt}}{S_{opt}} \quad (1)$$

For the independent set problem the equivalent measure is calculated by converting the size of the independent set found to the size of its associated vertex cover so that measures could be compared between the two conditions. From this value the associated PAO can be calculated.

PAO is not a valid measure for the Search condition in this experiment because a subject may decide that the given goal on an instance was not achievable and move on to the next instance. As a result, a skipped instance may not have a valid solution associated with it and therefore the size of the solution cannot be assumed to be valid. Instead the frequency of optimal solutions found across the 25 instances is used to compare performance between conditions.

Another measure of performance is the amount of search needed to find a solution. Both the number of moves needed as well as the amount of backtracking (or Undos) needed to find a solution are presented.

An indirect measure of performance is the amount of cognitive load experienced by participants while problem solving. Two measures of cognitive load are considered: the Time per Move and the Number of Toggles. The amount of time spent making a move selection can be seen as a measure of the amount of internal effort that is needed to select a move.

It was also observed that participants frequently toggled vertices on and off while trying to find a solution. While it is possible that these moves constituted legal additions or removal of vertices from the candidate solution, an alternate explanation is that they were making use of the user interface to test the impact of a move that was up for consideration.

A move was considered a Toggle if the currently selected vertex was selected in the previous move. Toggles could be an indication of increased cognitive load.

Analysis Tools and Data Collection

Bayes factors derived from standard analysis of variance were used to investigate evidence for or against the above-stated factors (Nickerson, 2000; Rouder, Morey, Speckman, & Province, 2012) for each of the listed measures. Bayes factor analysis was adopted for this analysis instead of null hypothesis significance testing (NHST) for a number of reasons. In scientific study it is typical to have the goal to evaluate the probability of a hypothesis given the observed outcome. NHST, however, only provides a way to evaluate the probability of the observed outcome (the data) given the null hypothesis (Nickerson, 2000), which is the inverse of what is typically desired. Bayes factor analysis, on the other hand, based on Bayes' theorem shown in Equation 2, provides a way to accept or reject either of two models, based on the relative posterior probabilities of competing models.

$$p(H \mid D) = \frac{p(D \mid H) \cdot p(H)}{p(D)} \tag{2}$$

Given two competing hypotheses, H_1 and H_2, the Bayes factor is computed to evaluate a change in prior odds based on the observed data. The relationship between the Bayes factor $\frac{p(D \mid H_1)}{p(D \mid H_2)}$, posterior odds $\frac{p(H_1 \mid D)}{p(H_2 \mid D)}$, and prior odds $\frac{p(H_1)}{p(H_2)}$ for each hypothesis is shown in Equation 3. Evidence for model H_1 against H_2 is defined according to Raftery (1995, p. 139), with a Bayes factor of 1–3 indicating weak evidence, 3–20 indicating positive evidence, 20–150 indicating strong evidence, and >150 indicating very strong evidence.

$$\frac{p(H_1 \mid D)}{p(H_2 \mid D)} = \frac{p(D \mid H_1)}{p(D \mid H_2)} \cdot \frac{p(H_1)}{p(H_2)} \tag{3}$$

To evaluate the impact of each of the factors considered in this work, Bayes factors for all models created by removing or leaving all main effects or their interactions from the full model were computed. JZS priors were used in the calculations, as suggested by Rouder et al. (2012). All analysis was conducted using R (R Project for Statistical Computing), and Bayes factors were computed using anovaBF in the R package developed by Morey, Rouder, and Jamil (2014).

RESULTS

In this study, a great number of performance measures were considered and analyzed in terms of the described factors and their interactions. For clarity, only those results with positive or stronger evidence are discussed in detail. Details of all measures can be found in Carruthers (2015).

GENERAL PERFORMANCE

Overall, participants found optimal solutions frequently with better performance in the Search condition of both problems. At the same time, more search was required in the Search condition indicating that this difference in performance comes with a cost in terms of the amount of effort needed to find a solution.

THE ROLE OF THE GOAL

To investigate how an infeasible-to-evaluate goal impacts performance on these problems, results are compared between the Optimization version, with an infeasible-to-evaluate goal, and Search version, with a feasible-to-evaluate goal. This comparison is done for both problems together (Problem Version factor) as well as with each problem considered separately (the interaction between Problem Version and Problem). As seen in Table 4, when both problems were considered together or separately, performance as measured by % Optimal and % Maximal/Minimal was better in the Search condition than in the Optimization condition. More search was required in the Search condition than in the Optimization condition, as measured by both the number of Moves, and the number of Undos. Bayes factor analysis found very strong evidence for the Problem Version factor on both performance measures (% Optimal and % Maximal/Minimal) with better performance in the Search version when the two problems are considered together. Positive evidence was found for the interaction between Problem and Problem Version as indicated by the % Optimal solutions found and very strong evidence was found for this interaction on the % Maximal/Minimal solutions found, as seen in Table 4. Similarly, very strong evidence was found for the Problem Version factor and the interaction between Problem and Problem Version factors on the number of moves, and positive evidence was found for the interaction between Problem and Problem Version on the number of Undos.

If a goal that is infeasible to encode results in goal modification, then it is expected that solutions to instances of problems where the goal is modified will better match solutions to a related problem with a goal that is feasible to identify than to the original, given, task. A number of goal modifications were proposed for the problems used in this study. To evaluate this hypothesis, performance is compared to solutions on problems which might be the result of goal modification.

The proposed modifications were predicted to impact performance in terms of % Optimal solutions, % Maximal/Minimal solutions, and the amount of search needed to find a solution. The locality of selections is also evaluated as an indication of the use of local strategies. Performance in the Optimization condition of each problem separately is of main

Table 4
Impact of infeasible-to-evaluate goal on performance. Bayes factor analysis evidence for this model also presented.

	Both problems			Independent set		Vertex cover		
Measure	**Search**	**Opt.**	**PVF**	**Search**	**Opt.**	**Search**	**Opt.**	**PPF**
% Optimal	76.89	48.95	***	83.75	57.23	70.13	40.67	*
PAO	NA	10.17	NA	NA	10.74	NA	9.60	NA
% Maximal/Minimal	95.40	82.29	***	95.00	91.00	96.00	74.00	***
Number of Moves	26.23	18.76	***	21.23	16.74	31.31	20.83	***
Adjusted Undos	15.17	7.041	***	9.72	5.97	20.73	8.15	*
Path length	1.769	1.919	**	1.99	2.05	1.55	1.79	*
Euclidean distance	174.5	184.3	*	176.17	177.63	172.71	191.05	*

Note. The column labeled PVF indicates the evidence for the Problem Version factor, and the column labeled PPF indicates the evidence for the interaction between Problem and Problem Version factors, with "***" indicating very strong evidence, "**" indicating strong evidence, "*" indicating positive evidence, "." indicating weak or no evidence. No evidence is given for PAO as no comparison can be made between Problem Versions.

interest because it is unlikely that goal modification would be identical on the two different problems. As shown in Table 4, participants in the Optimization version of the independent set condition found optimal solutions more than half the time, found maximal solutions with high frequency, and did little search. In contrast, in the Optimization version of the vertex cover problem participants in the Optimization condition found optimal solutions less than half of the time, found a moderate number of minimal solutions, and performed some search. Subsequent selections were closer as measured both by path length and Euclidean distance in the Search condition of both problems and in each problem separately. Path length or distance is measured as the number of edges that must be traversed to travel from vertex u to vertex v. The Euclidean distance between two vertices u and v is the distance *as the crow flies* between the vertices, or the length of the shortest line that connects them. There is strong evidence for the Problem Version factor on path distance and positive evidence for this factor on Euclidean distance. There is weak evidence for an interaction between the Problem Version and Problem factors on path distance and positive evidence for this interaction on Euclidean distance, as shown in Table 4.

Cognitive load is measured by the amount of time needed to make individual move selections and the amount of Toggles performed. In terms of the amount of time needed on average to make move selections, slightly more time was taken per move in the Optimization condition. In the independent set condition more time is taken per move in the Optimization condition, and in the vertex cover condition there is nearly no difference in the time taken per move between the Problem Versions. Positive evidence is found for the Problem Version factor alone and weak evidence is found of an interaction between Problem Version and Problem on this measure. Details can be seen in Table 5. Participants in the Search condition performed more Toggles with very strong evidence for the Problem Version factor on this measure and strong evidence for an interaction between Problem Version and Problem factors. Details can be found in Table 5.

DISCUSSION

Strong evidence was found for differences in performance on all major measures between goal well-definedness. This gives

Table 5
Impact of infeasible-to-evaluate goal on search and cognitive load. Bayes factor analysis evidence for this model also presented.

	Both problems			Independent set		Vertex cover		
Measure	**Search**	**Opt.**	**PF**	**Search**	**Opt.**	**Search**	**Opt.**	**PPF**
Time per Move	2.284	2.603	*	2.20	2.86	2.37	2.34	*
Toggles	7.899	3.94	***	6.11	3.96	9.70	3.93	**

Note. The column labeled PF indicates the evidence for the Problem factor, and the column labeled PPF indicates the evidence for the interaction between Problem and Problem Version factors, with *** indicating very strong evidence, ** indicating strong evidence, * indicating positive evidence, . indicating weak or no evidence.

preliminary evidence for the theory that the goals of the problem are encoded differently between these conditions. Similarly, the results support the theory that there may be performance differences, either between minimization and maximization problems, or between the vertex cover and independent set problems.

GENERAL PERFORMANCE

An original motivation of this work was to compare human performance on hard non-Euclidean optimization problems to that on other hard optimization problems like E-TSP. Results from the Optimization conditions of both the vertex cover and independent set conditions are in keeping with previously observed performance, with mean PAO close to 10% for both problems. Interestingly, however, performance on the Search versions of these same problems is *better* than the Optimization versions, measured as percent optimal solutions found. This raises the possibility that previously reported human performance results on E-TSP might be improved upon if participants are tasked with the Search version of that problem.

These general performance results suggest that visual processing alone is likely not responsible for the good quality previously noted on other hard optimization problems. The Gestalt principles, or the hierarchical pyramid model which have been proposed as being potentially partially responsible for solution quality, are likely not applicable on the non-Euclidean problems used in this study, and therefore cannot explain performance on the vertex cover and independent set problems.

IMPACT OF GOAL WELL-DEFINEDNESS

If participants in the Optimization condition are able to encode the goal exactly as given, then there should be no significant difference in performance between the Search and Optimization conditions of the same problem. This, of course, assumes that participants in the Search condition are able to encode the goal and are working on the task given. This possibility is investigated by considering how often optimal solutions are found in the Search condition. Participants are given the exact same instances to solve, with identical start states and legal operators, and the instances differ only in terms of the specificity of the given goal. If participants in the Optimization condition are unable to encode the given goal, encode it differently from in the Search condition, or encode some other goal, then it is expected that there will be a difference in performance between the two versions as a result of the difference in what constitutes a goal state between the two versions. Performance as measured by percent optimal solutions supports the theory that goals are encoded differently between these two versions: overall, performance by this measure is better on the Search version than on the Optimization version. Within each Problem condition, performance is also better on the Search version than on its associated Optimization version.

Another measure of performance is the amount of backtracking needed to find a solution. Differences in the amount of backtracking needed to find a vertex cover or independent set could be an indication of different goal states being encoded and searched for. Other possible explanations exist too, of course. The mean number of undos is much higher in the Search condition than in the Optimization condition, both across both problems, and within each problem. As shown in Table 4, this difference is much more pronounced in the vertex cover group, where the mean number of undos in the Search condition is over twice that in the Optimization version. One explanation for this difference is that relative to their associated Search versions, the problem being solved by participants in the Optimization condition of the vertex cover problem is much easier than that being solved in the independent set problem.

EVALUATION OF CANDIDATE GOAL MODIFICATION

With these findings in support of the hypothesis that participants in the Optimization condition are solving a problem other than the one given, the next task is to narrow down alternative problems that could explain performance. Four different goal modifications were identified for the minimum vertex cover and maximum independent set problems, which might explain performance, found in Tables 1 and 2 respectively. For each candidate modification, predicted results are compared to:

1. the quality of solutions found in the Optimization version,
2. the percent of minimal (VC) or maximal (IS) solutions found,
3. the percent optimal solutions found,
4. how much backtracking takes place.

These results can help narrow down which proposed modifications might explain performance on the Optimization versions of the problems given.

Minimal/Maximal Modification

One candidate modification is that of finding a minimal solution to the vertex cover problem or a maximal solution to the independent set problem. Recall that this modification predicts suboptimal solutions, many minimal/maximal solutions, few optimal solutions, and little backtracking. The PAO found in both problems supports this modification with the mean PAO near 10% in both problems. The percent optimal solutions found in the Optimization condition is also much lower than in the Search version, as predicted. And finally, very little backtracking is done in the Optimization condition which further supports this modification.

The percent of minimal/maximal solutions can be compared to the percent of optimal solutions to test which goal, optimization or minimal/maximal, better fits the data. These comparisons cannot prove that this restructuring modification is taking place; however, it can eliminate a candidate, or indicate that it is still a possibility. The percent of minimal solutions found in the Optimization version of the vertex cover problem is significantly higher than the percent optimal solutions in the same condition, and therefore is a better fit. These results strongly support the hypothesis that the infeasible-to-encode goal of the Optimization version of this problem could be restructured to that of a minimal solution. Similarly, the percent of maximal solutions found in the Optimization version of the independent set problem is significantly higher than the percent of optimal solutions in the same problem. Alternatively, it cannot be excluded that the problem could have been modified to another problem whose results coincidentally are also maximal/minimal with high likelihood.

Any Valid Cover/Independent Set

The second modification described discarded all sense of optimality altogether, and amounts to finding any valid vertex cover or independent set. Recall that this modification predicts suboptimal solutions, non-minimal/maximal solutions, few optimal solutions, and no backtracking. The PAO found in the two versions of both problems matches the prediction of suboptimal solutions in the Optimization version. In the vertex cover condition, the relatively lower mean percent minimal solutions may be an indication that this modification is taking place. As well, very few optimal solutions are found (fewer than 50%), which could be support for this modification. Finally, the number of undos deviates from the total lack of undos predicted for this modification. In the independent set condition, the high number of maximal solutions found in the Optimization condition does not match the predicted performance for this modification. The mean percent optimal solutions is slightly over 50%, but still well below the number found in the associated Search condition and also does not match this modification. Finally, the number of undos deviates from the total lack of undos predicted for this modification. In summary, there is mixed support for this modification for the vertex cover condition, and very little support for it in the independent set condition.

Local Optimization

Goal modification could also take the form of local optimization. Recall that this modification predicts suboptimal solutions, non-minimal/maximal solutions, few optimal solutions, and a moderate amount of backtracking. In the vertex cover condition, PAO matches the prediction of suboptimal solutions in the Optimization version, and the low percent optimal vertex covers found in the vertex cover condition supports this modification. The number of minimal vertex covers, however, is lower in this condition than in the independent set condition, and is weaker support for this modification. In the independent set condition PAO matches the prediction of suboptimal solutions in the Optimization version. In addition the high number of maximal independent sets found, and higher (over 50%) percent optimal independent sets found, supports this modification. Another possible measure that could support this modification is how local subsequent moves are. The average path distance between moves in all conditions is small, hovering around a length of two, which is further support of local optimization as a candidate modification. Interestingly, the closeness of subsequent moves is even more pronounced in the Search version than in the Optimization version, which suggests either that local choices drive move selections in these kinds of graph problems, independent of local optimization, or that local optimization is useful in attempting to solve both versions of these problems. In summary, mixed support is found for local optimization in both the vertex cover and independent set conditions.

Generation of Specific k*-Values*

The final modification proposed was the guess and check modification, which predicted near-optimal solutions, many minimal/maximal solutions, many optimal solutions, and an amount of backtracking similar to that seen in the Search conditions. The PAO in both problem conditions does not match the near-optimal solutions predicted by this modification. Similarly, the relatively low percent optimal vertex covers found in the vertex cover condition and independent sets found in the independent set condition does not match the optimality predicted. Finally, the low amount of undos in the Optimization versions of both problems relative to their associated Search versions also does not support this modification. These results do not support the likelihood that this modification takes place with significant frequency.

IMPACT OF GOAL MODIFICATION ON COGNITIVE LOAD

It is predicted that, when tasked with a problem with an aspect of the goal that is not encodable, problem solvers modify the goal to render it encodable. It was assumed that this modification should take place as parsimoniously as possible; that is, by modifying the problem as little as possible. It seems likely that this modification should also result in a problem that is not harder than an encodable version of the task (for example the Search version). However, it is important to note that this need not be the case. The Optimization version of the problem given could feasibly be converted into a problem that is at least as hard as the Search version of the respective problem, and still be encodable. Recall the proposed

modification in which increasingly smaller (or in the case of the maximization problem, independent set, larger) *k* values are tested until the optimal solution is found. If this modification took place, however, the number of moves needed to find a solution would be expected to be similar in the Optimization version to those used in the Search condition (which it is not), and for the percent of optimal solutions to be closer in the Optimization version to that found in the Search condition (which it is not). It is therefore assumed henceforth that goal modification results in a problem being encoded that is no harder than the one given.

One possible consequence of modifying the problem into an easier one is that it would likely take fewer moves to find a solution, and indeed this is what was observed in this study. In the Optimization version of both problems, the mean number of moves to find a solution was lower than that in the Search version. These results strongly support the hypothesis that participants in the Optimization condition are solving an easier problem. Participants in the Optimization condition consistently use fewer moves to find a solution, which could be an indication that the goal, whatever it might be, is easier to find than in the Search condition.

Toggles are likely an indication of using the interface to gain information about the current and near future states of the problem, then reduced cognitive load could manifest as fewer Toggles. Indeed, many fewer Toggles were observed in the Optimization condition than in the Search condition, overall, as well as for each problem considered separately (see Table 5). These results strongly support the hypothesis that participants in the Optimization condition solve a problem that is easier than the Search version, and that places less cognitive load on the system.

SUMMARY

Performance on the Optimization version of both the vertex cover and independent set problem is consistent with previously reported performance results on other hard optimization problems, namely E-TSP. This finding supports the hypothesis that performance on these non-Euclidean hard optimization problems would be similar to that found on Euclidean hard optimization problems, and suggests that problem-solving performance on this kind of hard optimization problem may not be solely dependent on visual processes that are only applicable to Euclidean problems. It is possible that problem solving on these non-Euclidean problems may still leverage, at some level, powerful visual processing mechanisms. But, it is not immediately clear how this might manifest itself.

Performance on the Search version of both the vertex cover and independent set problems is significantly better than that on the associated Optimization version, a finding that supports the hypothesis that the infeasible-to-evaluate goal of the Optimization version of these problems results in a different problem being encoded from that given. Cognitive load was also found to be higher in the Search condition than in the Optimization version, in support of the hypothesis that goal modification results in an easier problem being encoded than that given. Participants in the Search condition used far more search to find their solutions, found optimal solutions with higher frequency, and showed signs of greater cognitive load, all indications that the problem being encoded in the Search condition differed from that being encoded in the Optimization condition. This in turn implies that when tasked with the Optimization version of a hard computational problem, and facing a goal that is not encodable, problem solvers modify the problem and encode some other goal. This has important implications, not only in this work, but also in the interpretation of the results of previous studies which used hard optimization problems as instruments. How might performance on other hard problems differ if participants were tasked with the Search or Decision version? In this work, performance was found to be better on the Search version than on the Optimization version, and it is interesting to wonder if this might also be the case on hard problems like E-TSP.

Another important implication of these results relates to persistence and problem solving. In education, life, and work, people face many problems. Polya (1945) suggested that encouraging engagement with problem-solving tasks is important in order to develop good problem-solving skills. The significantly greater search observed in the Search condition of this study indicates that participants in this condition are much more persistent than those in the Optimization condition, that they are more engaged with the problem, despite working on a harder problem, under heavier cognitive load. The careful specification of goals may have educational implications, in particular in problem-solving education. In addition, while previous work concluded that goal specificity impeded problem solving (Sweller & Levine, 1982), the results of this study contradict this claim.

A framework for identifying candidate problem modifications was proposed, and based on this, four problem modifications were identified to try to explain what problem solvers tasked with the Optimization version of these problems might be solving. In the case of the minimum vertex cover problem, the greatest support was found for the minimal modification, suggesting that the non-encodable aspect, that is the optimality of the vertex cover, is replaced by the easier to evaluate goal of finding a minimal vertex cover. Similar results were found for the maximum independent set problem, with the maximal independent set problem best matching the results. This finding supports the hypothesis that an infeasible-to-encode goal is modified into a feasible-to-encode goal, although caution must be taken in interpreting these results. Four candidate modifications were proposed in this work and evaluated in

terms of a number of performance, search, and cognitive load measures. While by all accounts the maximal/minimal modification is the best fit for the results among those modifications identified, there are possibly many explanations for this match. The goal of either or both the given problems could have been encoded such that a different modification occurred, other than any identified here, for which maximal/minimal solutions are coincidentally likely.

The implications of this finding, that goal modification occurs when problem solvers are tasked with hard optimization problems, are non-trivial. What kinds of modifications might explain performance on other hard optimization problems, and how might that impact the interpretation of previous human performance results on hard optimization problems?

Modification of a problem with an ill-defined goal was predicted to result in a problem that is easier, one which results in lower cognitive load. The results of this study strongly support this prediction. Whatever goal modification(s) are taking place result in a problem-solving task which is significantly easier than the corresponding Search version.

CONCLUSION AND FUTURE WORK

This research contributes specifically to the study of human problem solving in three main ways. It refines our understanding of how people are able to cope with complexity, in particular when it may not be possible to know when a problem is solved. Second, it proposes a different approach for evaluating how people cope with complexity when they are free from being confounded by the challenge of determining when a solution is correct. Finally, the findings expand our understanding of how humans cope with complexity, presenting novel performance results for human problem solving of hard computational problems.

The main contributions of this work speak to improving the design of studies of human performance on hard computation problems and improving how performance results are interpreted on past and current studies that use these hard computational problems as instruments. Foundational to this contribution is the notion of gaining a sound understanding of how problem formulation impacts what problem can be encoded in the problem solver's internal representation, which in turn directly impacts what problem can, in fact, be solved.

COPING WITH ILL-DEFINED GOALS

Tasking problem solvers with instances of hard optimization problems has the potential to result in problem solvers being unable to encode the goal as given. Because the problem encoded in the inner representation defines the problem being solved, an inability to encode the given task implies that a different problem is being solved. This in turn has important ramifications when attempting to interpret the results of these studies. For one, the complexity results of the original task may no longer be representative of the complexity results of the problem being solved. In fact, if the results of this study are representative of how this mechanism manifests itself, the encoded problem is likely easier than intended. This also has implications when attempting to model performance.

A framework is provided for decomposing a problem's goal to identify what aspect of the goal is ill-defined, thereby making it possible to find candidate problems that might be encoded in place of the given optimization task. This framework was used, to serve as an example, to decompose the goal of the optimization version of the two problems used in this study. These candidate modifications were evaluated to determine what problems best explain performance.

How Does Goal Modification Take Place?

This research made no attempt to identify the mechanisms by which goal modification might take place, and instead focused on the problem at the computational level. But the question of what kinds of mechanisms might explain this process is deserving of investigation. It is very likely that goal modification as described here will be dependent on the nature of the task. For well-structured and otherwise well-defined problems like the computational problems typically found in the field of computational complexity, it may be possible, as done here, to decompose the problem's goals and identify candidate modifications. This is particularly true when working in a laboratory environment with subjects who are naive to the problem at hand. For real-world problems, however, the task of goal modification becomes more challenging, as complex factors like domain-specific knowledge and personal experience will impact how ill-defined goals are rendered well-defined ones.

Problem solvers do not appear to be aware of goal modification taking place, nor do they spend a great deal of time re-encoding the problem, suggesting that the underlying mechanisms that explain this process are *fast and frugal* (Newell, Weston, & Shanks, 2003). Furthermore, there is no reason to assume that goal modification occurs only once for a given problem or instance of a problem, or that it manifests itself identically between different problem solvers. It is likely that problem solvers adjust the encoding of the problem as they gain experience with it.

As a result, we find ourselves in a sea of unknowns if we wish to continue to use hard optimization problems as instruments in the study of human problem solving. Tasked with a problem with an ill-defined goal, problem solvers encode *some* problem, and may re-encode it *any number of times*, and experience and domain-specific knowledge will

likely impact the types of modifications that can take place. All that is known is that given a task with an ill-defined goal, the problem solver *cannot* be solving the given task, in general, which has a number of implications.

What Are the Implications of Goal Modification?

If computational complexity is taken into account, then the interpretation of human performance results is dependent on the problem being solved. This issue was not addressed in previous studies of human performance on hard optimization problems, which opens up a new way of interpreting the results of a number of studies. In Carruthers (2015), E-TSP and N-Puzzle were decomposed to isolate the component of the goal that is ill-defined; however, no candidate modifications were proposed. This is a great starting point from which to investigate what goal modifications might best explain performance on these problems, and to review previous attempts to model this performance.

What Are the Implications of These Findings in General?

The modification of ill-defined aspects of a problem has implications in other areas of problem-solving research and is not limited to the study of hard computational problems. For example, this approach is equally applicable in the study of decision-making or insight problems, and is closely related to explanations of performance in these areas. The difficulty of finding a solution to the nine dot problem, for example, has been attributed to problem solvers' inability to determine what constitutes a legal operator, rendering the set of legal operators ill-defined. Similarly, satisficing theory in decision making attempts to explain how people find solutions in a task environment rife with unknowns and uncertainties, by replacing the concept of *optimizing* with that of finding a solution that is *satisfactory.* Both these approaches try to explain performance by suggesting that problem solvers modify an aspect of the given problem. The difference in what we are presenting here is that in addition to a proposed modification, this work recognizes that the modification of any aspect of the given problem can, and often does, result in a modification of the task. This acknowledgement of the implications of goal (or other problem aspect) modification *at the computational level* makes it possible to better understand and evaluate algorithmic explanations for performance. There is potential to apply this same technique in other problem-solving research areas where ill-defined problem aspects are found.

It is worth noting, at this point, the difference between the performance impact of goal specificity, described earlier in this work, and the difference in performance based on the well-definedness of the goal. In maze tracing experiments, better performance was observed when the goal was not specified than when it was (Sweller & Levine, 1982). It is important to note, however, that the goal in both conditions of the maze tracing experiment was well-defined. Problem solvers could tell when the goal had been attained. In the research done in this work, in contrast, the non-specified goal of optimization renders the goal ill-defined, and therefore problem solvers are unable to determine if the goal has been attained. Therefore, while goal specificity was found previously to negatively impact problem solving in the context of maze tracing, it is not directly related to the performance results presented in this work.

COPING WITH COMPLEXITY IN PROBLEMS WITH WELL-DEFINED GOALS

A second main contribution of this work is the observation that it is possible to evaluate human performance on hard computational problems without the confounding factor of an ill-defined goal. As shown above, the Search version of hard optimization problems can be used to study how people navigate large problem spaces of hard problems, but with an encodable goal.

What Are the Implications of These Findings in General?

If the results of this study are representative of a general pattern in problem solving, then the better performance observed on instances of the Search versions of the problems used in this research implies that people may in fact be better at finding solutions to instances of hard computational problems than previously thought. This is a surprising result, one which was not expected, and one which warrants further investigation. This work does not discount the possibility that the surprisingly good performance results might be more an artifact of study design than an indication of how good people are at coping with complexity. Instance size and selection may have unintentionally resulted in an easier task than desired. However, performance on instances of the Search version problems used in this study is still better than that on those same instances of the associated Optimization version, raising the possibility that similar results might be found on other problems like E-TSP or N-Puzzle.

We note that in our study the instances presented to be solved for Search versions were stated such that the answer was always affirmative. Furthermore, the value searched for was always the best achievable value in the corresponding optimization version. We would like to expand this investigation of human performance of Search versions of intractable problems to instances where the given value is further away from optimal. It could be of particular interest to investigate the impact of selecting instances with values close to the optimum score that do not permit a solution.

Another observation is that the great deal of search that participants in the Search condition of both problems used in this study indicates a deep level of engagement with what are objectively complex problem-solving tasks. This has

important educational, workplace, and tool-design implications. As Polya (1945) stated, a key component in teaching problem solving is getting learners to engage with hard problems. If the results of this work are representative of the kind of engagement that can be elicited in naive problem solvers on these hard computational tasks, then visually presented tasks like these might prove to be useful tools in teaching problem solving. In the workplace and in the design of tools used for solving complex tasks, the use of concrete, encodable goals could also serve to encourage engagement with complex tasks. While it is known that data representation in general impacts problem-solving performance (Carroll, Thomas, & Malhotra, 1980; Marfil, Escolano, & Bandera, 2009; Norman, 2002; Noyes & Garland, 2003), the well-definedness of problem goals also clearly impacts performance. This knowledge can be leveraged to improve workplace patterns, by replacing ill-defined goals with well-defined alternatives to better represent complex tasks such that they can be meaningfully encoded in order to result in better engagement.

NOTES

[1] For NP-hard problems, no algorithms have been found to date that would determine solutions to instances of any finite size in a polynomial amount of time (here the polynomial has as parameter the size of the instance). The theory of NP-hardness (and NP-completeness) is developed for decision problems (problems that seek a YES or a NO as answer).

[2] A decision problem is NP-complete if any possible YES answer for a given instance can be verified in polynomial time.

[3] In this context, a *fast* algorithm is one which always yields a correct solution in time that is bounded by some polynomial function of the input size.

[4] In computational complexity, an approximation algorithm is one which produces a solution whose size can be guaranteed to be within a specific factor of optimal. Further, the time needed to find such a solution is typically *fast*.

[5] This representation may also include goal states that are not on direct paths to the goal, as participants need not always take direct routes to the goal.

[6] A more informal way to present this problem is to state that if the graph represents a social network where the vertices are people and the edges are relationships between people, then the goal is to find a largest group of people who do not know each other.

[7] When searching for a solution, problem solvers may consider more than one solution before deciding that one meets the given goal. These solutions are referred to as candidate solutions.

REFERENCES

Arora, S., & Barak, B. (2007). *Computational complexity: A modern approach*. New York, NY: Cambridge University Press.

Baddeley, A. D., & Hitch, G. (1974). Working memory. In *Psychology of learning and motivation* (pp. 47–89). New York, NY: Elsevier. https://doi.org/10.1016/S0079-7421(08)60452-1

Best, B., & Simon, H. (2003). Simulating human performance on the traveling salesman problem. *Proceedings of the Third International Conference on Cognitive Modeling* (pp. 1–6). Veenendaal, The Netherlands.

Carroll, J. M., Thomas, J. C., & Malhotra, A. (1980). Presentation and representation in design problem-solving. *British Journal of Psychology*, *71*, 143–153. https://doi.org/10.1111/j.2044-8295.1980.tb02740.x

Carruthers, S. (2015). *The role of the goal in problem solving hard computational problems: Do people really optimize?* (PhD thesis). University of Victoria, BC, Canada.

Carruthers, S., Masson, M., & Stege, U. (2012). Human performance on hard non-Euclidean graph problems: Vertex cover. *Journal of Problem Solving*, *5*(1), 34–55. https://doi.org/10.7771/1932-6246.1142

Downey, R. G., & Fellows, M. R. (2012). *Parameterized complexity.* London, UK: Springer.

Dry, M., Lee, M. D., Vickers, D., & Hughes, P. (2006). Human performance on visually presented traveling salesperson problems with varying numbers of nodes. *Journal of Problem Solving*, *1*(1), 20–32. https://doi.org/10.7771/1932-6246.1004

Dry, M., Preiss, K., & Wagemans, J. (2012). Clustering, randomness, and regularity: Spatial distributions and human performance on the traveling salesperson problem and minimum spanning tree problem. *Journal of Problem Solving*, *4*(1), 1–17. https://doi.org/10.7771/1932-6246.1117

Frisby, J. P., & Stone, J. V. (2010). *Seeing: The computational approach to biological vision.* Cambridge, MA: The MIT Press.

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

Gigerenzer, G. (2001). The adaptive toolbox. In G. Gigerenzer & R. Selten (Eds.), *Bounded rationality* (pp 37–50). Cambridge, MA: The MIT Press.

Gigerenzer, G., & Goldstein, D. G. (1996). Mind as computer. *Creativity Research Journal*, *9*(2, 3), 131–144.

Graham, S. M., Joshi, A., & Pizlo, Z. (2000). The traveling salesman problem: A hierarchical model. *Memory & Cognition*, *28*(7), 1191–1204. https://doi.org/10.3758/BF03211820

Karpinski, M., Lampis, M., & Schmied, R. (2013). New inapproximability bounds for TSP. In L. Cai, S. W. Cheng, & T. W. Lam (Eds.), *Algorithms and computation* (pp. 568–578).

Berlin, Germany: Springer. https://doi.org/10.1007/978-3-642-45030-3_53

Kong, X., & Schunn, C. D. (2007). Global vs. local information processing in visual/spatial problem solving: The case of traveling salesman problem. *Cognitive Systems Research*, *8*(3), 192–207. https://doi.org/10.1016/j.cogsys.2007.06.002

MacGregor, J. N., Chronicle, E. P., & Ormerod, T. C. (2004). Convex hull or crossing avoidance? Solution heuristics in the traveling salesperson problem. *Memory & Cognition*, *32*(2), 260–270. https://doi.org/10.3758/BF03196857

MacGregor, J. N., Chronicle, E. P., & Ormerod, T. C. (2008). A comparison of heuristic and human performance on open versions of the traveling salesperson problem. *Journal of Problem Solving*, *1*(1), 33–43.

MacGregor, J. N., & Ormerod, T. C. (1996). Human performance on the traveling salesman problem. *Perception & Psychophysics*, *58*(4), 527–539. https://doi.org/10.3758/BF03213088

MacGregor, J. N., Ormerod, T. C., & Chronicle, E. P. (2000). A model of human performance on the traveling salesperson problem. *Memory & Cognition*, *28*(7), 1183–1190. https://doi.org/10.3758/BF03211819

Marfil, R., Escolano, F., & Bandera, A. (2009). Graph-based representations in pattern recognition and computational intelligence. *IWANN*, 399–406.

Morey, R. D. (Maintainer), Rouder, J. N., & Jamil, T. (2014). Bayesfactor: Computation of Bayes factors for common designs. R package version 0.9.9.

Newell, A., & Simon, H. A. (1972). *Human problem solving.* Englewood Cliffs, NJ: Prentice-Hall.

Newell, B. R., Weston, N. J., & Shanks, D. R. (2003). Empirical tests of a fast-and-frugal heuristic: Not everyone "takes-the-best." *Organizational Behavior and Human Decision Processes*, *91*(1), 82–96. https://doi.org/10.1016/S0749-5978(02)00525-3

Nickerson, R. S. (2000). Null hypothesis significance testing: A review of an old and continuing controversy. *Psychological Methods*, *5*(2), 241–301. https://doi.org/10.1037/1082-989X.5.2.241

Norman, D. A. (2002). *The design of everyday things.* New York, NY: Basic Books.

Noyes, J. M., & Garland, K. J. (2003). Solving the tower of Hanoi: Does mode of presentation matter? *Computers in Human Behavior*, *19*(5), 579–592. https://doi.org/10.1016/S0747-5632(03)00002-5

Ormerod, T. C., & Chronicle, E. P. (1999). Global perceptual processing in problem solving: The case of the traveling salesperson. *Perception & Psychophysics*, *61*(6), 1227–1238. https://doi.org/10.3758/BF03207625

Pizlo, Z., Stefanov, E., Saalweachter, J., Li, Z., Haxhimusa, Y., & Kropatsch, W. G. (2006). Traveling salesman problem: A foveating pyramid model. *Journal of Problem Solving*, *1*(1), 83–101. https://doi.org/10.7771/1932-6246.1009

Polya, G. (1945). *How to solve it: A new aspect of mathematical method.* Princeton, NJ: Princeton University Press.

Raftery, A. E. (1995). Bayesian model selection in social research. *Sociological Methodology*, *25*, 111–163. https://doi.org/10.2307/271063

Rouder, J. N., Morey, R. D., Speckman, P. L., & Province, J. M. (2012). Default Bayes factors for ANOVA designs. *Journal of Mathematical Psychology*, *56*(5), 356–374. https://doi.org/10.1016/j.jmp.2012.08.001

Saalweachter, J., & Pizlo, Z. (2008). Non-Euclidean traveling salesman problem. In T. Kugler, J. C. Smith, T. Connolly, & Y.-J. Sun (Eds.), *Decision modeling and behavior in complex and uncertain environments* (pp. 339–358). New York, NY: Springer. https://doi.org/10.1007/978-0-387-77131-1_14

Simon, H. A. (1990). Invariants of human behavior. *Annual Review of Psychology*, *41*, 1–20. https://doi.org/10.1146/annurev.ps.41.020190.000245

Sweller, J., & Levine, M. (1982). Effects of goal specificity on means-ends analysis and learning. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, *8*, 463–474. https://doi.org/10.1037/0278-7393.8.5.463

Tak, S., Plaisier, M., & van Rooij, I. (2008). Some tours are more equal than others. *Journal of Problem Solving*, *2*(1), 4–28. https://doi.org/10.7771/1932-6246.1028

Todd, P. M., & Gigerenzer, G. (2000). Precis of simple heuristics that make us smart. *Behavioral and Brain Sciences*, *23*, 727–780. https://doi.org/10.1017/S0140525X00003447

van Rooij, I., Schactman, A., Kadlec, H., & Stege, U. (2006). Perceptual or analytical processing? Evidence from children's and adult's performance on the Euclidean traveling salesperson problem. *Journal of Problem Solving*, *1*(1), 44–73. https://doi.org/10.7771/1932-6246.1006

van Rooij, I., & Wareham, T. (2012). Intractability and approximation of optimization theories of cognition. *Journal of Mathematical Psychology*, *56*(4), 232–247. https://doi.org/10.1016/j.jmp.2012.05.002

van Rooij, I., Wright, C. D., & Wareham, T. (2012). Intractability and the use of heuristics in psychological explanations. *Synthese*, *187*(2), 471–487. https://doi.org/10.1007/s11229-010-9847-7

Vickers, D., Butavicius, M., Lee, M. D., & Medvedev, A. (2001). Human performance on visually presented traveling salesman problems. *Psychological Research*, *65*, 34–45. https://doi.org/10.1007/s004260000031