Submitted to ...

HOW WELL CAN HUMANS APPROXIMATE OPTIMALITY IN COMPUTATIONALLY HARD PROBLEMS?

NITIN YADAV

ANTHONY HSU

JUAN PABLO FRANCO

CARSTEN MURAWSKI
Department of Finance, University of Melbourne

PETER BOSSAERTS
Faculty of Economics, University of Cambridge

Many decisions people face are complex, even intractable. It is often suggested that decision-makers use heuristics to overcome complexity, and that the resulting behavior closely approximates optimality. The second claim deserves scrutiny because the theory of computation has concluded that solutions to certain classes of problems cannot reasonably be approximated. Still, the theory applies to algorithms of abstract computers – as opposed to humans – and classes are formed based on worst-case analysis. Here we report results of an experiment in which participants were asked to solve a number of tasks that differed in the degree of approximation complexity. We find that performance is as predicted by the theory. We elucidate how classification based only on worst cases can predict average human performance. Our findings demonstrate that properties of the task at hand explain performance without having to understand which heuristics are used, or which computational constraints humans face. (Version: August 2022)

Nitin Yadav: nitin.yadav@unimelb.edu.au
Anthony Hsu: anthony.hsu@unimelb.edu.au
Juan Pablo Franco: juan.franco@unimelb.edu.au
Carsten Murawski: carstenm@unimelb.edu.au
Peter Bossaerts: peter.bossaerts@unimelb.edu.au

KEYWORDS: Computational Complexity, Approximation Complexity, Traveling Salesperson Problem, Optimality, Experiment.

Model construction under these stringent conditions has taken two directions. The first is to retain optimization, but to simplify sufficiently so that the optimum (in the simplified world!) is computable. The second is to construct satisficing models that provide good enough decisions with reasonable costs of computation.

Herbert A. Simon, p. 498 in Simon (1979)

1. INTRODUCTION

Many economic models assume that decision-makers are always able to identify the most preferred option in a decision situation, appearing as if they always optimised their choices (Samuelson, 1948). It has been recognized that this assumption requires decision-makers to have inconceivable computational capacities (Veblen, 1908), sometimes referred to as "demonic powers" (Gigerenzer and Todd, 1999, Otworowska et al., 2018). While acknowledging that people do not possess unlimited computational capacities, it is often argued that decision-makers use heuristics to successfully approximate optimal behavior (Tversky and Kahneman, 1974, Gigerenzer and Selten, 2002). We will refer to this conjecture as the *Approximation Conjecture*. In this article, we examine both the theoretical and empirical plausibility of this conjecture in a context where ability to approximate has been formalized.

The context is one of difficult yet ubiquitous combinatorial problems, instead of the probabilistic setting that has inspired the "heuristics and biases" program in psychology (Tversky and Kahneman, 1974).[1] The reason is not only that these problems are very much

[1]We define a probabilistic setting as one where events are ergodic, so that objective probabilities can be obtained through repeated sampling. For combinatorial problems, without a subjective *a priori* distribution, it is impossible to imagine an experiment such that repeated sampling will establish definite probabilities as to whether any given instance of a combinatorial problem will have a solution (for decision problems – to be defined later) or as to the distribution of the extrema that can be obtained (for optimization problems – also to be defined later). See the discussion in Kolmogorov (1983).

a part of everyday life, but also because investigation of the anxiety associated with not knowing the correct answer has been much neglected as a source of uncertainty in human decision-making. In addition, ability to approximate the correct solution, and hence, to measure remaining uncertainty, has been formalized. Yet, the formalization applies to an abstract model of computers, which makes it relevant for electronic computers. Humans, instead, are biological computers, and as such is not clear whether the formalization extends. Through an experiment, we test whether this is the case, and if so, why.

Gilboa et al. (2021) have recently pointed out how one central problem in economics, the consumer problem, is hard. Indeed, it falls in the class of the combinatorial problems we study here. The authors suggest that consumers will resort to heuristics to approximate the optimum. One of the problems we study here is the weight-constrained shortest path problem, which is a variation on the consumer problem (formally known as the Knapsack Problem, KP). Our study can therefore be interpreted as providing an answer to the question: *how well* do human heuristics approximate the solution in the consumer problem?[2]

Our formalization draws on computational complexity theory (CCT), a theoretical framework for the analysis of the resources required to solve computational problems (Arora and Barak, 2009, Moore and Mertens, 2011). A key focus of CCT is to characterize which computational problems can be solved with bounded computational resources, in particular, time and memory, independent of the algorithm used. While this is the case for a large number of problems, an important insight of CCT has been that many problems, including many of great interest to economics, do not fall into this category. Resource requirements for these problems, which are typically referred to as *computationally intractable* (Garey and Johnson, 1979), are such that we cannot expect to be able to solve them within a reasonable amount of time. They include discrete optimization problems such as the knapsack problem (Kellerer et al., 2004) and constraint satisfaction problems

[2]Gilboa et al. (2021) also suggest that social interaction, in particular, imitation, may help individuals solve their consumption problem. Elsewhere, we have shown that markets can provide help (Meloso et al., 2009); when properly designed, markets can even be as effective as an "oracle" (Bossaerts et al., 2020).

such as the boolean satisfiability problem (Schaefer, 1978), along with many other problems of interest (Garey and Johnson, 1979, van Rooij et al., 2019).

Previous research has already linked human performance to classification of problems in terms of computational complexity. However, this work focused on reaching the exact solution, and in the cases where approximation considerations had been incorporated, these had been done in a problem-specific manner. Several studies have thus linked the quality of human problem-solving on instances of a single problem (e.g., 0-1 Knapsack Problem: Meloso et al. (2009), Bossaerts and Murawski (2017); TSP: MacGregor and Chu (2011); Sudoku: Pelánek (2011). More recently, problem-independent properties have been discovered that predicted human performance when instances are sampled in a specific random way (Franco et al., 2021b,a). Here, we extend this work by appealing to formal classification of problems in terms of how well solutions can be approximated.

In the following, we thus draw on the theory of approximation complexity (Ausiello et al., 2012, Wegener, 2005) to show that approximation of optimal solutions with bounded computational resources is only possible for a subset of problems. Approximation complexity refers to the computational resources required to approximate an (optimal) solution of a problem (Ausiello et al., 2012, Wegener, 2005). It has been shown that there exist problems where very good approximation of the optimal solution is possible with bounded computational resources (Ausiello et al., 2012). However, there also exist problems, again including many of relevance to economics, for which approximating the optimal solution is computationally as hard as computing the optimal solution itself. The latter set of problems include the maximum clique and maximum independent set problems, as well as the random version of the traveling salesperson problem (Ausiello et al., 2012).

We report the findings of a laboratory experiment in which we test if the quality of approximation of solutions suggested by human participants is related to the theoretical approximation complexity of the problem at hand. Specifically, we asked participants to solve instances of three similar computational problems, the weight-constrained shortest path problem, the euclidean traveling salesperson problem and the random traveling salesperson problem. These problems differ in their approximation complexity. We find that the approximation quality decreases with increasing levels of approximation complexity.

We then address the important question as to how a classification (of approximation complexity) based on worst-case analysis can shed light on average human performance of randomly selected instances. To do so, we rely on two considerations. First, the identity of the worst-case instance in general varies depending on the algorithm used, so that, if the algorithm is chosen at random each time a different instance is to be solved, in the worst case, one could end up with the worst approximation all the time. Second, we appeal to a previously discovered property, which is heterogeneity in solution approaches, over time and across humans (Bossaerts and Murawski, 2017). At the extreme, the heterogeneity could be modeled as if humans picked a feasible solution "at random." To emulate random variation of solution approaches, we investigate performance of a strategy that picked feasible solutions at random for the instances our participants had to solve. We focus on the variants of the traveling salesperson problem. We show that both the maximum error and the average error generates the same ranking across approximation classes as does human approximation performance. We hasten to add that humans perform substantially better than random search, confirming prior work (Bossaerts and Murawski, 2017).

Our contributions are three-fold. Firstly, we survey relevant results from approximation complexity theory; a framework that is able to explain human choice on optimization problems at a level that is finer than canonical notions of intractability (i.e., NP-Hard complexity class). Secondly, we provide experimental evidence that the theory of approximation complexity applies to human decision-making by showing that as approximation complexity increases, the quality of solutions computed by human decision-makers decreases. Thirdly, we provide a potential explanation why classification of problems based on worst-case analysis is relevant for the study of human average performance. Taken together, both the theoretical analysis and our empirical findings render implausible the aforementioned Approximation Conjecture. It certainly does not hold generically. CCT helps identifying situations where the Approximation Conjecture is expected to fail.

More studies on complexity have recently emerged in the economics literature. However, most concern very different types of complexity than the one we study here. Most prominently, Oprea (2020) studies what makes a rule/algorithm/heuristic complicated. In his case, the rule is given, and complexity of rule execution (i.e., procedural complexity) is the object of study. In our setting, the problem instance is given and a rule/algorithm/heuristic *can be chosen freely*. That is, we do not suggest how the instance is to be solved; we do not even require that the participant uses an algorithm in the formal sense of the term. Measures of actual procedural complexity in a real-world financial setting are reported in Colliard and Georg (2020). There exists work on economies as complex systems, but this work too is of a very different nature: it concerns nonlinearities in the dynamics of aggregate quantities that characterize the economy, in the limit producing chaotic processes; see, e.g., Arthur (2018). In contrast, the complexity of finding the best allocations in mechanism design and auction theory is of the same nature as the complexity we study here, but concerns the designer or auctioneer, not the agents; see, e.g., Roughgarden and Talgam-Cohen (2019). Granted, there is an emerging literature on learning by participants in auctions, but this does not involve computational complexity, but sample complexity, i.e., the notion that a sample may not be diverse enough to learn, with high probability, the correct rule when only a finite number of observations are available; see, e.g., Nedelec et al. (2022).[3] Attempts to measure the complexity of real-world decision problems such as health insurance choice have started to emerge in economics journals. For instance, armed with evidence that insurees often choose dominated health insurance contracts (Bhargava et al., 2017),[4] with an experiment, Beşliu (2022) determines which aspects of the contracts cause participants to make most mistakes.[5]

[3]Sample complexity metrics have been developed for the machine learning community; see, e.g., Kakade (2003).

[4]Bhargava et al. (2017) show that the simplest metric of complexity, “menu complexity” (the number of options and features), does not explain mistakes.

[5]She shows that coverage similarity – akin to correlation between values and costs in a Knapsack Problem – and state complexity – the number of payoff-relevant states, akin to the number of items in a Knapsack Problem –

2. APPROXIMATION COMPLEXITY CLASSES

The theory of computational complexity provides a formal way to categorize problems into easy and hard classes. A problem is deemed easy if the time required to solve it scales at most polynomially in the problem size. An algorithm is called tractable if the number of steps it takes scales polynomially in the size of the input. In contrast, a problem is hard if the time required to solve it (with the best known algorithm) scales superpolynomially in the problem size. In general, computational complexity caters to specific type of problems distinguished based on their output, namely, decision and optimization problems. The output for a decision problem is either “yes" (a solution exists) or “no" (a solution does not exist). The problem of checking if a path exists that visits each vertex only once in a given graph is an example of a decision problem. In contrast the output for an optimization problem is a solution that minimizes (or maximizes) a given objective value function while satisfying given constraints. For an minimization problem, a feasible solution is one that meets all the specified constraints, and an optimal solution is a feasible solution that minimizes the value of the objective function.

For hard optimization problems (NP-hard; see Appendix A.1 for a formal definition of hard problems) finding an optimal solution may require the decision maker to expend an impractical amount of time. Hence, one may seek algorithms that provide a balance between the quality of the solution (i.e., how close can one get to the optimal value) and the time taken to produce a solution. Approximation complexity provides a formal framework to establish limits on these guarantees, thus providing guidance on how well can a problem be approximated.

The notion of a problem being “as hard as” another problem is crucial to establishing equivalence between problems that belong in the same complexity classes. The notion of at-least-as-hard is characterized via a reduction argument. Simply put, a problem B is at-least-as-hard as a problem A if A can be solved by a set of tractable operations plus referring to a solution of B. In this case we say that A can *be reduced* to B. For instance, it can

significantly affect choice quality, while the number of elementary computations necessary to compute the plans' expected payoffs does not.

be shown that the decision variant of the traveling salesperson problem can be reduced to its optimization variant. Consider an instance in which the optimal path length is k^* and where the corresponding decision variant asks to determine the existence of a path to visit all cities with length less or equal to a distance k. The decision variant can be solved by simply solving the optimization variant and comparing k^* with k. The answer is “Yes” if and only if $k^* \leq k$. Note that this reasoning is indifferent to the algorithm that was used to solve the problem (e.g., we did not make any assumptions on how we computed k^* as the minimal path length).

Importantly, the theory behind problem reduction provides generalizability to our work. If one can solve a hard problem by only using tractable operations, then one can also solve all problems that can be reduced to it by using only tractable operations. The number of problems that can be reduced to canonical hard decision problems are indeed large (e.g., see Ruiz-Vanoye et al. (2011), and the class NP-Complete Arora and Barak (2009)). Hence, we expect our findings to be representative of a class of problems that share similar characterstics to the ones chosen in this study.

2.1. *Approximation complexity*

It is an open question whether a polynomial time algorithm exists for all problems in the class NP. Hence (as of this writing) one cannot expect to find optimal solutions for NP-Hard optimization problems in polynomial time. However, one could change the question about goodness of an algorithm, to: how close can one get to the correct solution if one only has a polynomial amount of time available? The theory of approximation complexity provides an answer.

In the presence of bounded resources, we assume that the time available to the decision-maker to solve a problem, is a polynomial function of the size of that problem. In approximation complexity theory the main objective is to categorize problems into complexity classes based on how well tractable algorithms can guarantee to approximate optimal solutions. This approximation guarantee is generally estimated in terms of bounds on the realtive performance. Without loss of generality we will restrict the definitions of approximation complexity classes to minimization problems (for details see Ausiello et al. (2012), Moore and Mertens (2011)).

DEFINITION 1: Let w be a feasible solution for an instance x of an minimization problem, and w^* is an optimal solution for x. *Relative performance* is defined as:

$$R(x, w) = \frac{\mathsf{value}(w)}{\mathsf{value}(w^*)},$$

where $\mathsf{value} : \{w | w \text{ is a feasible solution}\} \to N^+$ is a given objective function that maps a feasible solution to a number.

We require that there exists at least one feasible solution and that the values of all feasible solutions are positive. For minimization problems relative performance is always greater than equal to 1. Given two feasible solutions w_1 and w_2 with relative performance values as r_1 and r_2, respectively, $r_1 < r_2$ implies that w_1 is of better quality (that is, a better approximation) than w_2. An optimal solution will have a relative performance value of 1.

Based on the notion of relative performance one can define algorithms that compute approximate solutions. Utilizing this approximation metric one can measure goodness of approximation algorithms in terms of guarantees on the approximation quality.

DEFINITION 2: An algorithm $\mathcal{A}$ is a *r-approximate* algorithm for an minimization problem if for all instances x of the problem, the relative performance is bounded by a given constant r, that is, $R(x, \mathcal{A}(x)) \leq r$, where $\mathcal{A}(x)$ is the feasible solution obtained by executing $\mathcal{A}$ on instance x.

One can combine the notion of an approximation algorithm with the bounds on available resources (i.e., time) to get finer segregations of NP-Hard optimization problems. The first class we define requires existence of an approximation algorithm for any arbitrary bound.

DEFINITION 3—APX, Approximable: An NP-Hard minimization problem belongs to the complexity class APX if there exists a polynomial time r-approximate algorithm for it where $r \geq 1$. We call such a problem as an approximable problem.

The class APX contains NP-Hard optimization problems that allow for algorithms that can compute solutions with some quality guarantees. However, there are NP-Hard problems for which this is not possible (unless P=NP). These problems are said to be inapproximable.

DEFINITION 4—Inapproximable: An NP-Hard minimization problem is inapproximable if there does not exist any polynomial time r-approximate algorithm for it.

In other words, for inapproximable problems approximating is as hard as finding an optimal solution, whereas for approximable problems one can obtain approximate solutions in polynomial time. However, note that for approximable problems the bound on goodness is arbitrary.

Problems in the class APX can be further refined based on the relationship between their efficiency and the bound r of an approximation algorithm. APX optimization problems that allow tractable approximation algorithms to be built for any given r belong to class *PTAS* (polynomial time approximation scheme).

DEFINITION 5—PTAS: An NP-Hard minimization problem is in the class PTAS if there exists an algorithm $\mathcal{A}$ that takes two inputs, an instance x of the problem and a desired relative performance ratio r, returns a r-approximate solution for x in time that is polynomial in the size of x.

Problems that belong to PTAS allow a decision maker to choose the quality of approximation. However, note that the definition of class PTAS is subtle. The requirement on polynomial time is only on the size of input, whereas the algorithm takes two inputs (instance x and the desired ratio r). Therefore, it may well be the case that the algorithm runs in time exponential to $1/(r-1)$ (minimum time needed could be proportional to $|x|^{1/(r-1)}$, for instance, where $|x|$ denotes the size of the instance x). As such, time could increase exponentially as the desired performance ratio increases.

Finally, PTAS problems that allow approximate algorithms that scale well both to the instance size and performance ratio belong to the class *FPTAS* (fully polynomial time approximation scheme).

DEFINITION 6—FPTAS: An NP-Hard minimization problem is in the class FPTAS if there exists an algorithm $\mathcal{A}$ that takes two inputs, an instance x of the problem and a desired relative performance ratio r, returns a r-approximate solution for x in time polynomial in the size of x and in $1/(r-1)$.

Taken together, in terms of easiness of approximation, problems in class Inapproximable cannot be approximated, followed in decreasing order, by APX, PTAS and FPTAS (note that both PTAS and FPTAS are a subset of the APX class). Next, we describe the three graph-based minimization problems that we chose as example members of the approximation complexity classes.

2.2. *Traveling salesperson and weight-constrained shortest path problems*

Random TSP and Euclidean TSP: The traveling salesperson problem (TSP) involves finding a minimal cost cyclic route that visits each city. In TSP, one is given a set cities $C = \{1, \ldots, n\}$ and a $n \times n$ square matrix d where $d_{i,j}$ is the distance (cost) to travel from city i to city j. A route is a permutation $\pi : C \to C$ where $\pi(i) = j, i \neq j$ implies that city j should be visited after city i. A route is cyclic if it revisits the initial starting city. The cost of a route π is the sum of costs incurred by traveling between cities while following π. A solution to TSP is a cyclic route π that visits all cities and minimizes $\sum_{i \leq n} d_{i,\pi(i)}$. We consider the symmetric version of the problem, in which the cost to travel between two cities does not depend on the direction of travel (that is, for all cities i and j it is the case that $d_{i,j} = d_{j,i}$). If one had to use brute force (by listing all feasible city combinations) then the size of the search space would grow factorially in the number of cities. The best known algorithm to solve TSP correctly requires time that grows exponentially in the number of cities.

Different variants of the TSP exist that differ on the restrictions imposed on the distance matrix d. In the most general case, when there are no restrictions placed on d, TSP is inapproximable. That is, guaranteeing a bound on the performance ratio is as hard as finding an optimal solution. An example of this is the Random TSP (RTSP) where the distances are sampled uniformly from a given range. If one restricts the distance matrix d to contain only

Euclidean distances (Euclidean TSP — ETSP) the problem is approximable and belongs to the class PTAS (Ausiello et al., 2012).[6]

Weight-constrained shortest path problem: The weight-constraint shortest path problem (WCSPP) involves finding the shortest path that satisfies a weight constraint. In our case we will replace the notion of weight with cost. The problem consists of a set of cities $C = \{1, \ldots, n\}$, a start city s, and end city e, a natural number W, and two functions $d : C \times C \to \mathbb{N}$ and $w : C \times C \to \mathbb{N}$ where $d(i, j)$ encodes the distance between cities i and j and $w(i, j)$ denotes the cost of travel between i and j. The objective of the problem is to find a path $\pi = p_1 \cdots p_k$ such that $p_1 = s$ (path starts in city s), $p_k = e$ (path ends in city e), $\sum w(p_i, p_{i+1}) \leq W$ (the path cost is at most W), and π minimizes $\sum d(p_i, p_{i+1})$. [7] WCSPP is in the class FPTAS, which is the class of hard problems that are easiest to approximate. Algorithmically, WCSPP is easier to approximate than TSP because the inclusion of a constraint allows search to be restricted based on feasibility (regions of the search space that obey the cost constraint).

3. METHODS

Instance generation: Instances for the three problems were generated based on graphs. Explicitly, an instance of each of the three problems considered is represented in a graph where nodes are cities and edges are times (and/or costs) between cities. Each city (i.e., node) is associated with an x,y-coordinate. In the ETSP and RTSP the edges represent the travel time between cities. In the WCSPP, each edge has two values: travel time and cost. The number of nodes was fixed at 10 for all instances of the three problems. The coordinates of a city were generated by sampling uniformly at random from a 1000×1000 grid. Graphs for RTSP and ETSP were fully connected (each instance had 45 edges) whereas graphs for WCSPP were half connected (each instance had 23 edges). Since each edge in WCSPP had twice the amount of information (cost and time) as compared to TSP (time), this ensured that the amount of information displayed to the participants across

[6]Euclidean distances satisfy triangular inequality: $d(i, j)^2 \leq d(i, j')^2 + d(j', j)^2$, all i, j, j'.

[7]Note that WCSPP is similar to the Knapsack problem (KP)

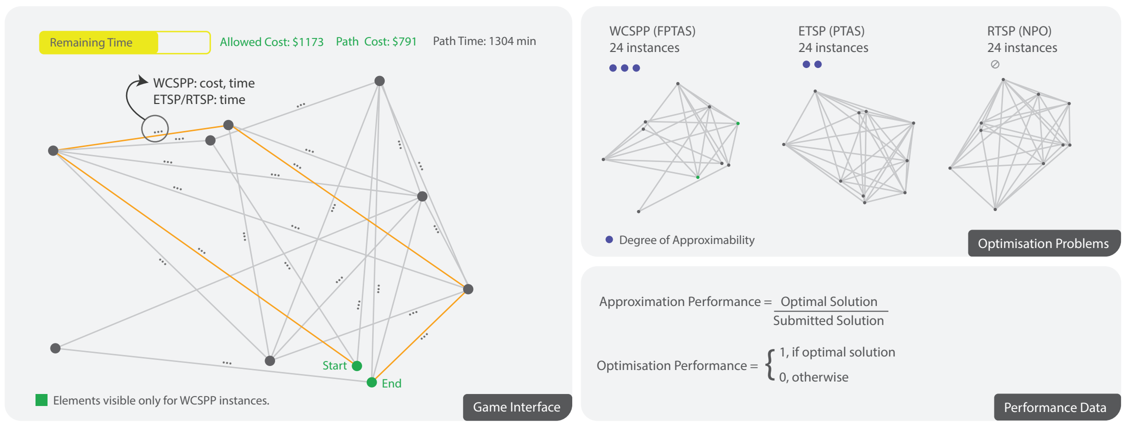


FIGURE 1.—Overview of experimental paradigm.

Game Interface: RTSP, ETSP and WCSPP were shown as interactive graphs. Time remaining and instance specific information was displayed at the top area of the interface. Optimisation Problems: Each participant had to solve 24 instances for each problem. All instances had 10 number of vertices. Performance Data: For each attempt the quality of submitted solution was recorded.

problems was comparable. Graphs in WCSPP were chosen uniformly at random from all graphs with 10 nodes and 23 edges. Table A.I summarizes the instance properties.

In the ETSP the time (i.e., edge) between cities is given by the euclidean distance between cities based on its x,y-coordinates. In the RTSP, the travel time between two cities was generated by uniformly sampling from $[1, 1000]$ irrespective of the x,y-coordinates of the cities. In the case of WCSPP, the cost and time travel between two cities was sampled from the same range of $[1, 1000]$. The coordinates of the cities shown in ETSP underwent a second randomization. This was done to minimize the effect of visual representation on problem solving (MacGregor and Chu, 2011, van Rooij et al., 2006). Participants were made aware that the visual coordinates of cities were independent of the travel time between them.

In the WCSPP, for each sampled graph different instances were generated with different combinations of the start city, the end city, and the total cost constraint. All instances that were unclear visually on the game graphical interface were excluded (e.g., instances where the difference between coordinates of two cities was not sufficient to view the edge information clearly). For RTSP and ETSP 24 instances were randomly chosen from the remaining instances. For WCSPP instances whose optimal solution was a direct connection between start and end city were also excluded. This adds to the robustness of our results since including these instances would have made the average performance of WCSPP higher. The remaining instances were binned based on the length of solution. This resulted in three bins that had instances with optimal solution length of 3, 4 and 5. The proportion of instances with solution length of 3, 4 and 5 was 81.82%, 17.19%, and 0.99%, respectively. 24 WCSPP instances were then randomly selected so that they were representative of the population in terms of the length of the solution. Out of these 24, 19 instances had a solution length of 3, 4 instances had a solution length of 4, and 1 instance had a solution length of 5.

Participants and experimental task: Twenty-four participants (age range=$[18, 35]$, mean age=22.4 years, 12 female, 12 male) were recruited from the general population. Inclusion criteria were based on age (minimum=18 years, maximum=35 years) and normal or corrected-to-normal vision. The experimental protocol was approved by the University of Melbourne Human Research Ethics Committee (Ethics ID 1852087.1). Written informed consent was obtained from all participants prior to commencement of the experimental sessions. Experiments were performed in accordance with all relevant guidelines and regulations, including the Declaration of Helsinki.

The experiment included three problems, RTSP, ETSP and WCSPP. These three problems belong to the NP-Hard class in terms of finding an optimal solution, but differ in their approximation complexity. In terms of approximation difficulty, RTSP is inapproximable and, as such, we hypothesized it would be the most difficult problem. This problem is followed by the ETSP which is an APX problem in PTAS. Finally, the easiest problem to approximate would be the WCSPP, which is another APX problem that is not only in PTAS, but also in FPTAS. Hence, amongst the three problems, WCSPP is theoretically the easiest to approximate, followed by ETSP and RTSP.

Participants were asked to solve 24 instances each of WCSPP, ETSP, and RTSP (72 instances per participant, see Figure 1). The order in which instances were presented to a participant was randomized within each problem. The order of problems was randomized as well. Participants were given a maximum of 90 seconds to solve each instance and they were given a break of 15 minutes between each problem. Prior to the actual session, each participant completed a 45-minute practice session. To make the task incentive-compatible, participants received a payment proportional to the quality (i.e., closeness to the optimal value) of their submitted solutions (between $\$0$ and $\$1$ per instance). In addition, participants received a show-up fee of $\$8$.

Instances were presented as a graph-based computer game in which participants built a solution by clicking the vertices of a graph to generate a path. Instance parameters (maximum cost, time and cost of the current solution for WCSPP, and current cost for RTSP and ETSP) were displayed on the top of the screen (see game interface panel in Figure 1). The time remaining in an instance to submit a solution was indicated by a bar at the top left of the screen. The game functionality allowed participants to undo prior moves as well as to reset the current selection (i.e., remove all path selections).

Data analysis: For each trial of the task, we recorded the ID of the participant, the time taken for that instance, the submitted solution, and how close the value of the submitted solution was to the optimal value of the instance. We also recorded each mouse click that resulted in either selecting or deselecting a vertex, and whether the participant used the reset-selection functionality. To calculate the optimal value of each instance, we encoded the instance in Minizinc and solved it using the Gecode constraint solver. Data analysis was performed in Python (using the Pandas and SciPy packages) and R.

4. RESULTS

We use two metrics to measure participant's performance. *Approximation performance* is defined as the ratio $\mathsf{value}(w^*)/\mathsf{value}(w)$ where w^* is an optimal solution and w is the submitted solution. Since our problems are minimization problems, the approximation performance will be in $(0, 1]$. Given two approximation performance values r_1 and r_2, $r_1 > r_2$

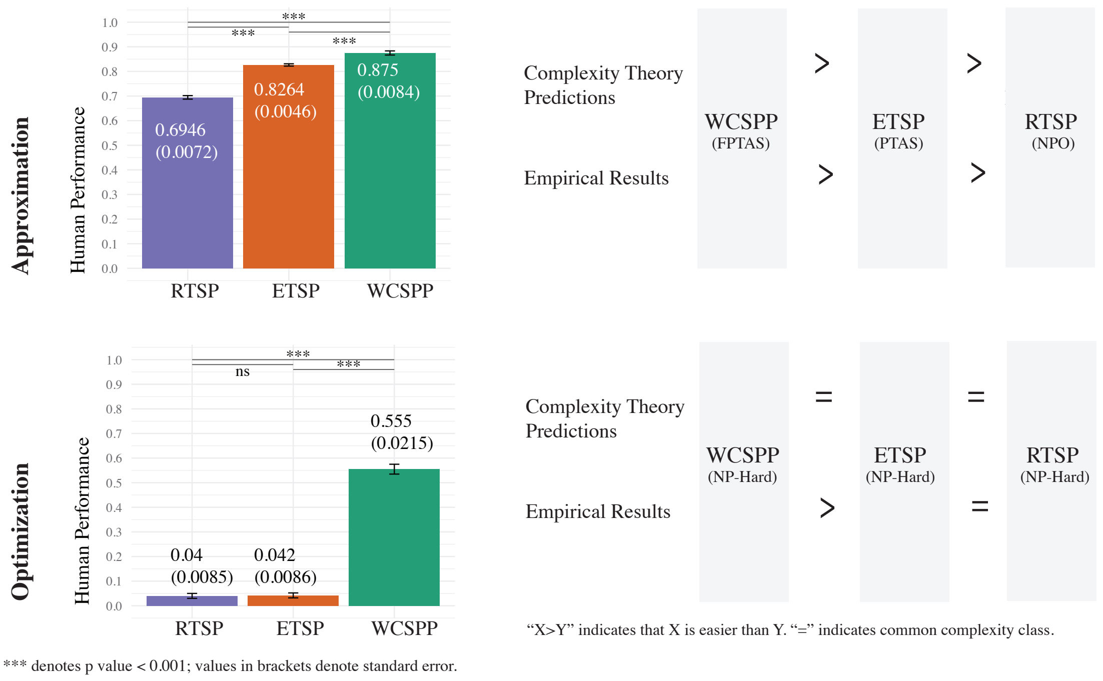


FIGURE 2.—Approximation and Optimization performance for WCSPP, ETSP, and RTSP.

indicates that r_1 had a better (that is, a higher value) solution than r_2. An approximation performance value of 1 indicates that an optimal solution was submitted. Note that the approximation performance metric is the inverse of relative performance (see Section 2). We use the approximation performance metric to report our results as it is easier to interpret. It is on a scale that is directly related to a participant's approximation performance. Theoretically, we expect approximation performance to be highest for WCSPP, followed by ETSP, and lowest for RTSP (see Figure 2, right Panel). The second metric, *Optimization performance*, is binary variable that takes a value of 1 if an optimal solution was submitted, and 0 otherwise. Since our instances are for problems that all belong to the class NP hard, theory predicts that, in terms of reaching the optimum, they are equivalent. That is, from the point of view of finding the optimal solution, the theory does not differentiate. We indicate this in Figure 2, Right Panel, by means of equality signs.

In total 93.98% solutions $(1624$ data points) were submitted before the time limit of 90 seconds. The mean time on task for valid submissions for the WCSPP, ETSP and RTSP was 51.97s (SE=1.01s), 65.03s (SE=0.69s), 62.37 (SE=0.73s), respectively (see Table B.I in SOM for the summary statistics). The fastest submission was for WCSPP (6.73s). To evaluate the performance data we excluded the attempts that ran out of time and therefore, did not submit a solution.

4.1. *Both approximation and optimization performance decreased with approximation complexity*

The mean approximation performance (see Figure 2) was highest for WCSPP (0.87, SE=0.008), followed by ETSP (0.83, SE=0.0046), and RTSP (0.69, SE=0.007). The mean of the optimization performance was highest for WCSPP (0.56, SE=0.0215), followed by equal means for ETSP (0.04, SE=0.0086) and RTSP (0.04, SE=0.0085). For all the three problems there was always an instance and a participant who found an optimal solution.

To accommodate heterogeneity, we used a mixed effects model to predict performance from the problem type (see Table B.II in SOM for details). In all the models participants were included as random intercepts to allow for inter-individual variability. We used a linear linkage function with Gaussian likelihood to model the approximation performance and a logit linkage function with binomial likelihood to model the optimization performance.

The mean approximation performance decreases by 0.048 (SE=0.009) from WCSPP to ETSP (significant at 0.001), and by higher value of 0.179 (SE=0.09) from WCSPP to RTSP (significant at $p < 0.001$). Mean approximation performance decreases by 0.131 from ETSP to RTSP (significant at $p < 0.001$).

In terms of the optimization performance, 55.5% of attempts yielded an optimal solution for WCSPP. However, optimization performance decreased significantly, by 51%, when comparing WCSPP and the two TSP variants (significant at $p < 0.001$). The difference in the optimization performance between ETSP and RTSP was not significant ($p > 0.05$; see Figure 2).

We also analyzed the performance metrics at the individual level. The difference in mean approximation performance for WSCPP-ETSP, WSCPP-RTSP and ETSP-RTSP was significant in 54.2%, 87.5% and 79.16% of participants, respectively. At the participant level, 100% displayed higher mean optimization performance for WCSPP than ETSP (significant at $p \leq 0.05$), and 95.8% of the participants had higher mean optimization performance for WCSPP than RTSP (significant at $p \leq 0.05$). We did not find any significant difference in mean optimization performance between ETSP and RTSP. Importantly, we did not find any significant difference of mean approximation and optimal performance in the direction opposite to what computational complexity theory predicts (see Table B.III and Table B.IV in SOM for details).

4.2. *Effort increased in approximation complexity*

We use the number of valid mouse clicks and the time spent on an instance as proxies for participant effort. A mouse click is deemed valid if it results in a change in the current working solution (e.g., adding a city, removing a city, or resetting the current solution). The average number of clicks and the average time spent on WCSPP instances was 9.07 and 54.8s, respectively. Both of these proxies of effort increase from WCSPP to TSP variants (significant at $p < 0.001$). The average number of clicks increased by 3.98 from WCSPP to ETSP and by 4.6 from WCSPP to RTSP, whereas the average time spent increased by 11.48s from WCSPP to ETSP and by 9.09s from WSCPP to RTSP (see Table B.V in SOM for details). The difference between ETSP and RTSP on the number of clicks and time spent was only weakly significant ($p < 0.05$). Although participants had a higher number of clicks on RTSP instances than ETSP, they spent more time on ETSP instances than RTSP instances.

4.3. *Productivity decreased with approximation complexity*

We define productivity as approximation performance per unit of effort. We consider two metrics for productivity, approximation performance per unit of time spent, and approximation performance per mouse click. Productivity decreased as approximation complexity increased (see Figure B.4 in SOM). Specifically, participants displayed an approximation performance of 0.156 per mouse click and 0.024 per second for WCSPP. The approximation performance decreased by 0.089 per mouse click and 0.01 per second for ETSP, and by 0.101 per mouse click and 0.011 per second for RTSP (all results are significant at $p < 0.001$). Productivity was higher for ETSP than for RTSP (significant at $p < 0.001$ for approximate performance per click and at $p < 0.05$ for approximate performance per second). See Table B.VI in SOM for details.

4.4. *A mechanism for worst-case analysis to make valid predictions on randomly selected instances*

As mentioned in the Introduction, approximation complexity classes are based on worst-case analysis, while we studied average performance on random instances of three classes. Because the identity of the worst case depends on the algorithm used, and humans exhibit enormous diversity in solution strategies (across instances and individuals), we set out to determine whether a random-selection strategy would generate the effect we observe among humans. Specifically, we investigated whether the maximum possible error from randomly selecting a feasible solution differed across problem variants. We also studied whether the average possible error of this strategy is related to approximation complexity. Here, we focus our attention on the two TSP variants due to underlying structural differences between the TSP and the WCSPP that could confound this analysis (see SOM B.4 for details and for the WCSPP results).

We first construct the feasible solution space for the instances. For TSP the feasible solution space includes all cyclic paths that visit each city once. Each entry in the feasible solution space represents a feasible solution. An optimal solution is a feasible solution with the minimum value. The quality of a feasible solution can be measured in terms of how far it is from the optimal, or in other words, how much error a decision maker would have made if they submitted that particular feasible solution. An error for a feasible solution w is defined as $1 - \mathsf{value}(w^*)/\mathsf{value}(w)$ where w^* is an optimal solution (for details see SOM B.4).

We investigated the distribution of these errors when considering the performance over the space of feasible solutions. We computed two metrics for the errors: the expected maximum error, computed by averaging across instances the maximum possible error on each instance (that is, the error of the feasible solution whose value is farthest from the optimal value), and the expected average error, computed by first averaging errors for an instance across feasible solutions, and then taking the mean of all the averages. The expected maximum error and the expected average error were 0.778 and 0.628 for RTSP and 0.588 and 0.449 to ETSP, respectively. The differences in both metrics were significant at $p < 0.001$. As conjectured, the errors are higher for RTSP than for ETSP.

These results do not imply that participant choices were random. To the contrary, participants performance was better than a random algorithm. Random selection algorithms would predict an average human performance of 0.372 for RTSP and 0.551 for ETSP, respectively, whereas human performance was 0.695 and 0.826, respectively. In addition, participants had spent effort differentially on RTSP and ETSP (see Section 4.2) whereas a random algorithm would spend the same effort on both TSP variants. Overall, when these results are taken together with prior findings that human solution strategies exhibit enormous variability across instances and individuals (Bossaerts and Murawski, 2017), they suggest that the effect of approximation complexity on human performance is driven partially, but not entirely, by the heterogeneity in solution approaches among humans.

5. CONCLUSION

The Approximation Conjecture states that humans will not be able to make optimal choices, but will deploy heuristics that allow them to approximate optimality well. Here, we put this to a test in a context where we hoped we could appeal to results of the theory of approximation complexity, a sub-field of Computational Complexity Theory (CCT). This theory predicts that there exist problems where the Approximation Conjecture fails, and humans will not be able to approximate optimality well. Inherent properties of the problem at hand are sufficient to identify when this happens; there is no need to analyze what the problem solver is “thinking.”

We find that as approximation complexity increases, humans are farther from the optimum even while effort increases. Additionally, we show that solution performance (the chance of actually finding the optimal solution) decreases. These results support the premise that approximation complexity classes predict human behavior on intractable (NP-hard) problems. We have established here that, contrary to the Approximation Conjecture, quality of approximation may be poor. At the same time, CCT predicts when this happens.

We have focused on approximation complexity classes. Still, there are other, related dimensions of computational hardness that could be employed to predict approximation quality, such as the size of the instance (number of nodes in a TSP) or problem representation (graphical *vs.* list representation of TSP). Future work could explore the effect of these properties on human performance and, in particular, on approximation quality.

By studying random feasible solutions to instances of the TSP, we propose a resolution to the puzzle as to why a classification based on worst-case analysis is capable of predicting average performance. We discovered that approximation complexity classes also predict performance of random selection algorithms. We therefore postulate that the well-documented heterogeneity in solution approaches among humans (Murawski and Bossaerts, 2016) ensures that analysis based entirely on worst cases (the theory) has value for analysis of average performance (the observation), since the worst case changes with the specific algorithm chosen.

The results reported here and in the earlier literature (cited in the Introduction) show that theoretical categorization of problems in terms of crude classes such as P *vs.* NP can be refined in order to better capture variability in human performance. The three problems we studied belong to the class NP-hard, and hence, in principle, are equivalent in terms of difficulty of finding an optimal solution (see equal signs, Right Panel, figure 2). Yet performance, effort and productivity are very different. Markedly, participants did not know that they were attempting two different variants of the traveling salesperson problem. However, despite spending similar time on the two variants, approximation performance was significantly different. Importantly, since participants were not explicitly told about the differences in terms of computational complexity across the three problem variants, it is unlikely that our results emerged because of deliberate reduction of productivity as approximation complexity increases.

One might wonder whether our results are surprising. If one assumed that human decision-making was computational ("algorithmic"; see Chalmers (2011), van Rooij et al.

(2019)), then our results should be expected. However, it is still an open question whether human cognition is actually purely computational. Still, we were able to link average human performance to a classification based on computational difficulty in terms of length of time algorithms need to approximate the solution. Humans appear to choose *as if* they deploy algorithms, just like they choose in simple settings with perfectly divisible goods *as if* they maximize some utility function (Crawford and De Rock, 2014).

Our findings imply that the degree to which optimality can be approximated depends not only on the computational resources of the decision-maker, but also on the hardness of the underlying cognitive problem at hand. They demonstrate that human performance can be predicted from problem properties independently of the nature of the heuristics actually used. Most work on the effect of complexity on human behavior has focused on characterizing the heuristics humans use in the face of complexity and how this leads to biases. That is, the emphasis has generally been on "what are people thinking?" following Newell et al. (1972). This approach is difficult to generalize across problems and people because, in the context of combinatorial problems, humans evidently use a variety of approaches (Murawski and Bossaerts, 2016). Nevertheless, our findings are like those in probabilistic tasks. There too, one can understand human choice to a large degree by simply focusing on instance properties. For instance, in the multi-armed bandit problem, features of the bandit at hand, such as payoff variability, shed light on when humans stop exploring and, simultaneously, how well they perform (Payzan-LeNestour and Bossaerts, 2015).

With this study, we have thus established that "what problems are people solving?" matters to predict performance. This is important, for two reasons. First, theoretically, it supports the idea that *a priori* plausibility of cognitive models has to be based, in part, on features of the problem at hand, and in particular, the complexity class it belongs to (van Rooij et al., 2019, Frixione, 2001, van Rooij and Wareham, 2012). Second, from a practical point of view, our framework could be used to set standards on how to present choices in order to reduce decision mistakes, even in the absence of information as to how people decide. Presentation of health insurance and retirement investments, work schedules, budgets, web site navigation, etc. could thus be guided, in part, by measuring approximation complexity. Preferably, the presentation should be such that it induces instances of problems in the class FPTAS.

APPENDIX A: PROBLEM HARDNESS AND COMPUTATIONAL COMPLEXITY CLASSES.

A.1. *What are hard problems?*

The notion of “hard” problems is at the core of this work. We use the theory of computational complexity (Ausiello et al., 2012, Moore and Mertens, 2011) to segregate problems into hard and easy categories. We say a problem is *hard* if it belongs to the class NP-hard. In order to explain this characterization two concepts require introduction: problems and algorithms.

We consider two types of problems, namely *decision* and *optimization* problems. An optimization problem is one whose objective is to optimize (i.e., maximize or minimize) a given function while adhering to certain constraints. For example, the objective in the optimization variant of the traveling salesperson problem is to minimize the cost of a path with the constraint that the path should be a cycle (that is, the path should end in the same city as the starting city) that visits all cities. In contrast, a decision problem is one whose answer is either “Yes” or ‘No”. For example, the objective in the decision variant of the traveling salesperson problem is to determine whether there exists a path of length less than a given threshold that visits all the cities. One can view an algorithm for a decision problem as a function that decides if the given instance belongs to a Yes-category (i.e., solutions is “Yes”) or a No-category (i.e., solutions is “No”). For an instance of a decision problem an algorithm often provides what is called a *witness* for the instance. The witness is a proof or certificate of the answer. For example, given a path P for the decision variant of the traveling salesperson problem, one can compute its total distance traveled and check against the given threshold. If the distance traveled is less than the threshold, then the solution is “Yes” (i.e., the instance belongs to the Yes-category) and the path P is a witness of the solution. One way to solve a problem is through an *algorithm*, a sequence of computational steps that provides an answer to any instance of the given problem. An algorithm uses computational resources (e.g., time and memory) to solve a particular instance. Note that, under this definition, the answer does not have to be correct.[8].

We are interested in the question: how good is an algorithm? One definition of the goodness of an algorithm is in terms of the amount of resources it uses in relation to the size of the problem. One such metric is to measure how much the time taken by an algorithm scales in the worst case (that is, the instances that take the longest to solve) as the problem size increases. A problem is called *tractable* (that is, it can be solved in a reasonable time) if there exists an algorithm that finds the *correct* answer and whose time grows at most *polynomially* in the size of the problem. In such a case we say that the algorithm solves the problem in polynomial time.

Problems that have similar resource requirements (in terms of algorithms that solve them) can be grouped together into *computational complexity classes*. A decision problem is in class P if there exists an algorithm that solves it correctly in polynomial time. On the other hand a decision problem is in class NP if a witness for a "Yes"-instance can be checked in polynomial time (irrespective of how long it took to solve the instance). The class NP includes problems for which there does not (yet) exist a polynomial time algorithm. Such problems are called *intractable* as the best algorithm known for them takes more than polynomial amount of time (e.g., some algorithms take time that grows exponentially in the problem size).

Expanding on the NP class, the class NP-hard is defined. A problem (decision or optimization) is said to be in the class NP-Hard if it is *at-least-as-hard* as any other problem in NP. The notion of at-least-as-hard is characterized via a reduction argument. Simply put, a problem B is at-least-as-hard as a problem A if A can be solved by a set of tractable operations plus referring to a solution of B. In this case we say that A can *be reduced* to B. For instance, it can be shown that the decision variant of the traveling salesperson problem can be reduced to its optimization variant. Consider an instance in which the optimal path

[8]To be consistent with the definition of approximation algorithms, we do not require that an algorithm should always provide a correct answer. An algorithm is *sound* if the answer it provides is correct. An algorithm is *complete* if it finds all correct answers.

length is k^* and where the corresponding decision variant asks to determine the existence of a path to visit all cities with length less or equal to a distance k. The decision variant can be solved by simply solving the optimization variant and comparing k^* with k. The answer is "Yes" if and only if $k^* \leq k$. Note that this reasoning is indifferent to the algorithm that was used to solve the problem (e.g., we did not make any assumptions on how we computed k^* as the minimal path length).

Instance generation: Instances for the three problems were generated based on graphs. Explicitly, an instance of each of the three problems considered is represented in a graph where nodes are cities and edges are times (and/or costs) between cities. Each city (i.e., node) is associated with an x,y-coordinate. In the ETSP and RTSP the edges represent the travel time between cities. In the WCSPP, each edge has two values: travel time and cost. The number of nodes was fixed at 10 for all instances of the three problems. The coordinates of a city were generated by sampling uniformly at random from a 1000×1000 grid. Graphs for RTSP and ETSP were fully connected (each instance had 45 edges) whereas graphs for WCSPP were half connected (each instance had 23 edges). Since each edge in WCSPP had twice the amount of information (cost and time) as compared to TSP (time), this ensured that the amount of information displayed to the participants across problems was comparable. Graphs in WCSPP were chosen uniformly at random from all graphs with 10 nodes and 23 edges. Table A.I summarizes the instance properties.

In the ETSP the time (i.e., edge) between cities is given by the euclidean distance between cities based on its x,y-coordinates. In the RTSP, the travel time between two cities was generated by uniformly sampling from $[1, 1000]$ irrespective of the x,y-coordinates of the cities. In the case of WCSPP, the cost and time travel between two cities was sampled from the same range of $[1, 1000]$. The coordinates of the cities shown in ETSP underwent a second randomization. This was done to minimize the effect of visual representation on problem solving (van Rooij et al., 2006). Participants were made aware that the visual coordinates of cities were independent of the travel time between them.

In the WCSPP, for each sampled graph different instances were generated with different combinations of the start city, the end city, and the total cost constraint. All instances

Problem	**Number of Cities**	**Number of Edges**	**Edge information size**	**Average metric between cities**
R(andom)TSP	10	45	45	Average time: 525.37
E(uclidean)TSP	10	45	45	Average time: 488.02
W(eight)C(onstrained)SPP	10	23	46	Average time: 527.64 Average cost: 477.08

The average time and cost denotes their means across edges. For TSP each edge cost is denoted in terms of travel time. For WCSPP each edge had a travel time and a travel cost.

TABLE A.I

SUMMARY OF INSTANCE PROPERTIES FOR RTSP, ETSP AND WCSPP.

that were unclear visually on the game graphical interface were excluded (e.g., instances where the difference between coordinates of two cities was not sufficient to view the edge information clearly). For RTSP and ETSP 24 instances were randomly chosen from the remaining instances. For WCSPP instances whose optimal solution was a direct connection between start and end city were also excluded. This adds to the robustness of our results. The remaining instances were binned based on the length of solution. This resulted in three bins that had instances with optimal solution length of 3, 4 and 5. The proportion of instances with solution length of 3, 4 and 5 was 81.82%, 17.19%, and 0.99%, respectively. 24 WCSPP instances were then randomly selected so that they were representative of the population in terms of the length of the solution. Out of these 24, 19 instances had a solution length of 3, 4 instances had a solution length of 4, and 1 instance had a solution length of 5.

APPENDIX B: RESULTS

We use two metrics to measure participant performance. *Approximation performance* is defined as the ratio $\mathsf{value}(w^*)/\mathsf{value}(w)$ where w^* is an optimal solution and w is the submitted solution. Since our problems are minimization problems, the approximation performance will be in $(0, 1]$. Given two approximation performance values r_1 and r_2, $r_1 > r_2$ indicates that r_1 had a better (that is, a higher value) solution than r_2. An approximation performance value of 1 indicates that an optimal solution was submitted. Note that the approximation performance metric is the inverse of relative performance (see Appendix **??**). We use the approximation performance metric to report our results as it is easier to interpret. It is on a scale that is directly related to a participant's approximation performance. The second metric, *Optimization performance*, is binary that takes a value of 1 if an optimal solution was submitted, and 0 otherwise.

TABLE B.I

SUMMARY STATISTICS FOR PERFORMANCE AND TIME ON INSTANCES OF WCSPP, ETSP AND RTSP.

	WCSPP (N = 533)	ETSP (N = 547)	RTSP (N = 544)
Approximation performance			
minimum	0.21	0.51	0.17
median (IQR)	1.00 (0.79, 1.00)	0.84 (0.75, 0.91)	0.70 (0.58, 0.81)
mean (sd)	0.87 ± 0.19	0.83 ± 0.11	0.69 ± 0.17
se	0.0084	0.0046	0.0072
maximum	1.00	1.00	1.00
Optimization performance			
minimum	0	0	0
median (IQR)	1 (0.00, 1.00)	0 (0.00, 0.00)	0.00 (0.00, 0.00)
mean (sd)	0.56 ± 0.50	0.04 ± 0.20	0.04 ± 0.20
se	0.0215	0.0086	0.0085
maximum	1	1	1
Time			
minimum	6.73	20.42	20.84
median (IQR)	53.16 (32.22, 70.71)	66.11 (53.12, 77.45)	64.24 (48.89, 76.34)
mean (sd)	51.97 ± 23.34	65.03 ± 16.12	62.37 ± 16.96
se	1.0108	0.6894	0.7270
maximum	90.00	90.00	90.00

In total 93.98% solutions were submitted before the time limit of 90 seconds. The mean time on task for valid submissions for the WCSPP, ETSP and RTSP was 51.97s (SEM=1.01s), 65.03s (SEM=0.69s), 62.37(SEM=0.73s), respectively (see Table B.I for the summary statistics). The fastest submission was for WCSPP (6.73s). To evaluate the performance data we exclude the attempts that ran out of time and therefore, did not submit a solution. The mean of the approximation performance was highest for WCSPP (0.87, SEM=0.008), followed by ETSP (0.83, SEM=0.0046), and RTSP (0.69, SEM=0.007). The mean of the optimization performance was highest for WCSPP (0.56, SEM=0.02), followed by equal means for ETSP (0.04, SEM=0.0086) and RTSP (0.04, SEM=0.0085). For all the three problems there was always an instance and a participant who found an optimal solution.

B.1. *Approximation and optimization performance*

We used a mixed effects model to regress performance from the problem type (see Table B.II for details). In all the models participants were included as random intercepts to allow variability across participants. We used a linear linkage function with gaussian likelihood to model the approximation performance and a logit linkage function with binomial likelihood to model the optimization performance. We note that although one would expect a non-infinitesimal mass at the region where performance is 1, the kernel density estimates (Figure B.1) of the approximation performance displays errors (relative to the predicted mean) for ETSP and RTSP whose mode is substantially below 1. This is not the case for WCSPP. Still, the errors exhibit a long left tail. This justifies the use of the gaussian model in a quasi-maximum likelihood estimation of the model parameters.

The mean approximation performance decreases by 0.048 (SE=0.009) from WCSPP to ETSP (significant at 0.001), and by a higher value of 0.179 (SE=0.09) from WCSPP to RTSP (significant at $p < 0.001$). Mean approximation performance decreases by 0.131 from ETSP to RTSP (significant at $p < 0.001$). The direction of change in the approximation performance across the problems fits the prediction from computational complexity

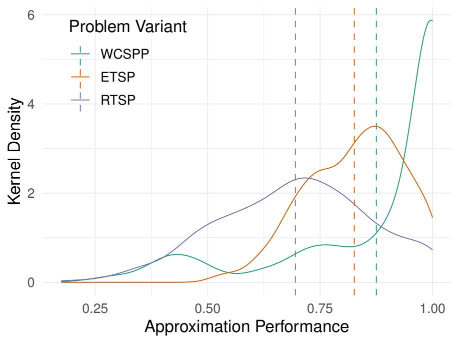
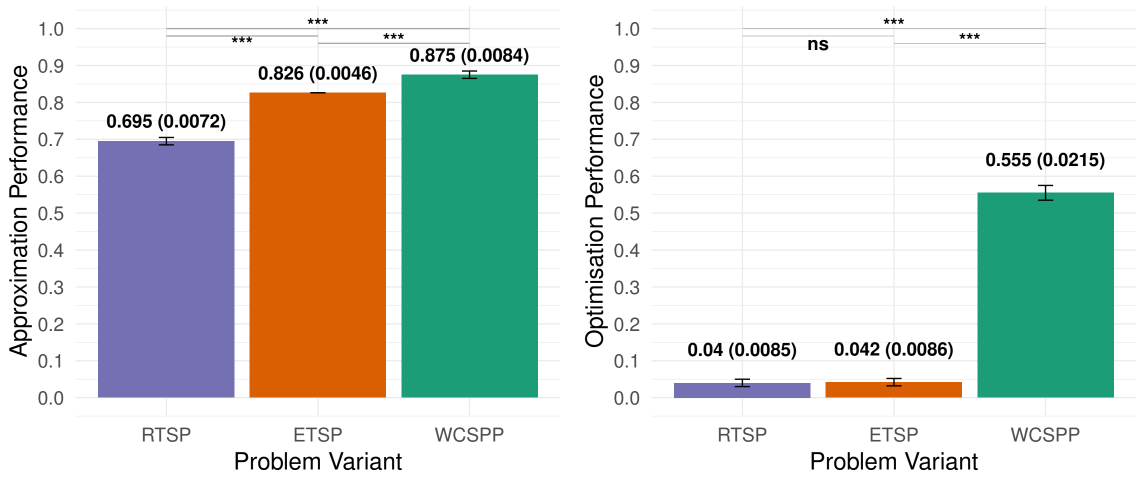


FIGURE B.1.—Kernel density estimate for approximation performance.

theory, namely: approximation performance is inversely related to approximation complexity (see Figure B.2).



*** denotes p value less than 0.001; Values in brackets denote standard error.

FIGURE B.2.—Mean approximation performance for WCSPP, ETSP, and RTSP.

In terms of the optimization performance, 55.5% of attempts yielded an optimal solution for WCSPP. However, optimization performance decreased significantly, by 51%, when comparing WCSPP and the two TSP variants (significant at $p < 0.001$). The difference in the optimization performance between ETSP and RTSP was not significant ($p > 0.05$; see Figure B.2). According to computational complexity theory, finding an optimal solution is equally hard for ETSP and RTSP, even though one may be easier to approximate for ETSP. Our empirical results accord with this prediction. So, while mean approximation performance in ETSP is higher than in RTSP, the difference in the mean optimization performances between ETSP and RTSP is not significant.

TABLE B.II

MIXED EFFECTS MODEL FOR PERFORMANCE WITH RANDOM INTERCEPTS TO ACCOMMODATE PARTICIPANT HETEROGENEITY.

	Dependent variable:	
	Approximation performance	Optimization performance
	(1)	(2)
	Estimates (Std. Error)	Odds Ratios (Std. Error)
Intercept (WCSPP)	0.874^{***}	1.250
	(0.011)	(0.016)
ETSP	-0.048^{***}	0.0328^{***}
	(0.009)	(0.020)
RTSP	-0.179^{***}	0.0317^{***}
	(0.009)	(0.020)
Random effects (Participant, $n = 24$)		
σ^2	0.024	3.290
ICC	0.065	0.040
Observations	1,624	1,624
Marginal R^2 / Conditional R^2	0.182 / 0.235	0.432 / 0.454
Note:		*p<0.05; **p<0.01; ***p<0.001

Table B.III shows the differences at participant level in mean approximation and optimization performance for the three problems, WCSPP (W), ETSP (E), and RTSP (R). For the comparison between approximation performance, a “✓" symbol in the table indicates that the performance difference between the relevant problem pair is significant (at $p \leq 0.05$) and in the right direction (as predicted by computational complexity theory), “–" indicates that the difference was not significant ($p > 0.05$). We did not find any significant difference of mean approximation performance in the direction opposite to what computational complexity theory predicts. The difference in mean approximation performance for WSCPP-ETSP, WSCP-RTSP and ETSP-RTSP was significant in 54.2%, 87.5% and 79.16% of participants, respectively. There was a high difference in mean optimization performance between WCSPP and the TSP variants (also see Table B.IV). At the participant level, 100% displayed higher mean optimization performance for WCSPP than ETSP (significant at $p \leq 0.05$), and 95.8% of the participants had higher mean optimization performance for WCSPP than RTSP (significant at $p \leq 0.05$). In line with computational complexity theory, we did not find any significant difference in mean optimization performance between ETSP and RTSP. A value of 0 in Table B.III indicates that the participant did not find an optimal solution for any TSP instance. Table B.IV lists, by participant, the number of instances where an optimal solution was found.

B.2. *Effort*

We use the number of valid mouse clicks and the time spent on an instance as proxies for participant effort. A mouse click is deemed valid if it results in a change in the current working solution (e.g., adding a city, removing a city, or reseting the current solution). Table B.V shows the result for two mixed effects models, with the number of mouse clicks and the time spent as the dependent variable, the problem variant as the independent variable, and participant as random intercept. The average number of clicks and the average time spent on WCSPP instances was 9.07 and 54.8s, respectively. Both of these proxies of effort increase from WCSPP to TSP variants (significant at $p < 0.001$). The average number of clicks increased by 3.98 from WCSPP to ETSP and by 4.6 from WCSPP to TSP, TABLE B.III

APPROXIMATION AND OPTIMIZATION PERFORMANCE DIFFERENCES AT PARTICIPANT LEVEL.

Id	Approximation			Optimisation		
	W ≠ E	W ≠ R	E ≠ R	W ≠ E	W ≠ R	E ≠ R
1	✓	✓	✓	✓	✓	–
2	✓	✓	✓	✓	✓	–
3	✓	✓	–	✓	✓	–
4	✓	✓	✓	✓	✓	0
5	–	✓	✓	✓	✓	0
6	✓	✓	✓	✓	✓	–
7	✓	✓	–	✓	✓	–
8	✓	✓	✓	✓	✓	0
9	✓	✓	✓	✓	✓	–
10	–	–	–	✓	✓	–
11	–	✓	✓	✓	✓	0
12	–	–	–	✓	–	–

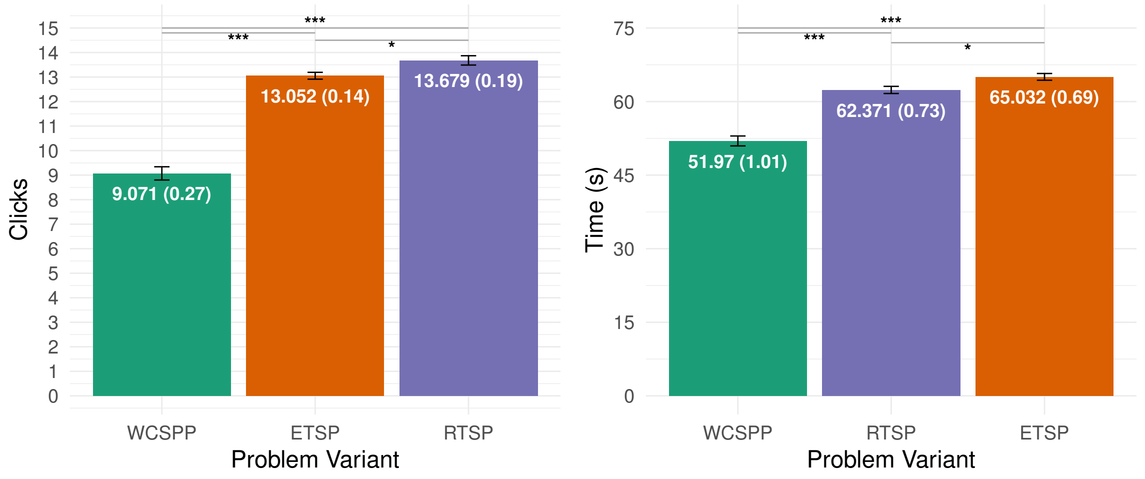
Id	Approximation			Optimisation		
	W ≠ E	W ≠ R	E ≠ R	W ≠ E	W ≠ R	E ≠ R
13	✓	✓	✓	✓	✓	–
14	–	✓	✓	✓	✓	–
15	✓	✓	✓	✓	✓	–
16	✓	✓	✓	✓	✓	–
17	✓	✓	–	✓	✓	–
18	✓	✓	✓	✓	✓	–
19	–	✓	✓	✓	✓	–
20	–	✓	✓	✓	✓	–
21	–	–	✓	✓	✓	–
22	–	✓	✓	✓	✓	–
23	–	✓	✓	✓	✓	–
24	–	✓	✓	✓	✓	–

TABLE B.IV

NUMBER OF INSTANCES WHERE A PARTICIPANT FOUND AN OPTIMAL SOLUTION.

Id	WCSPP	ETSP	RTSP	Id	WCSPP	ETSP	RTSP	Id	WCSPP	ETSP	RTSP
1	13	2	0	9	12	3	2	17	17	0	1
2	17	3	1	10	8	0	2	18	14	0	1
3	15	0	1	11	9	0	0	19	8	1	0
4	13	0	0	12	10	2	4	20	9	1	0
5	9	0	0	13	19	1	2	21	11	3	1
6	14	1	1	14	8	0	1	22	9	1	1
7	15	0	1	15	16	2	2	23	13	2	0
8	13	0	0	16	18	0	1	24	6	1	0

FIGURE B.3.—Mean effort (number of clicks and time in seconds) for WCSPP, ETSP, and RTSP.



* denotes p-value less than 0.05; *** denotes p-value less than 0.001; Values in brackets denote std. error.

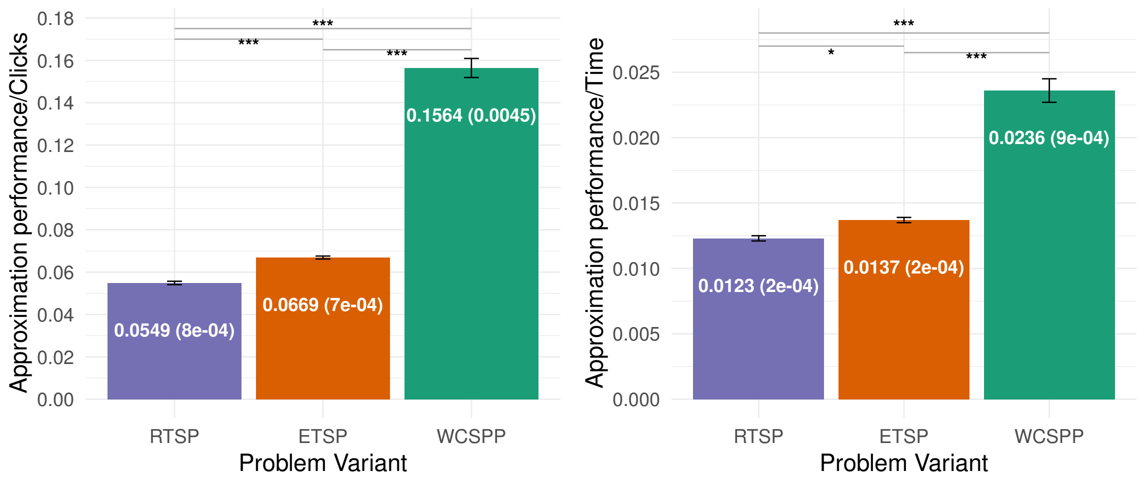
whereas the average time spent increased by 11.48s from WCSPP to ETSP and by 9.09s from WSCPP to RTSP. The difference between ETSP and RTSP on the number of clicks and time spent was only weakly significant ($p < 0.05$). Although participants performed a

TABLE B.V

MIXED EFFECTS MODEL FOR EFFORT WITH PARTICIPANT-LEVEL RANDOM INTERCEPTS.

	Dependent variable:	
	Number of mouse clicks	Time spent in seconds
	(1)	(2)
Fixed effects [Estimate (Std. Error)]		
Intercept (WCSPP)	9.071***	54.809***
	(0.558)	(2.304)
ETSP	3.981***	11.480***
	(0.252)	(0.998)
RTSP	4.608***	9.097***
	(0.252)	(0.998)
Random effects (Participant, $n = 24$)		
σ^2	18.276	286.619
ICC	0.065	0.015
Observations	1,728	1,728
Marginal R^2 / Conditional R^2	0.143 / 0.373	0.057 / 0.328
Note:		*p<0.05; **p<0.01; ***p<0.001

higher number of clicks on RTSP instances than ETSP instances, they spent more time on ETSP instances than RTSP instances (see Figure B.3).



* denotes p-value less than 0.05; *** denotes p-value less than 0.001; Values in brackets denote std. error.

FIGURE B.4.—Productivity for WCSPP, ETSP, and RTSP.

B.3. *Productivity*

We define productivity as approximation performance divided by the amount of effort exerted. We consider two metrics for productivity, approximation performance per unit of time spent (Seconds), and approximation performance per mouse click. Table B.VI shows the details of the mixed effects models for the relationship between productivity and problem variants. Productivity, as measured by the two metrics, decreased as approximation complexity increased (see Figure B.4). Participants displayed an approximation performance of 0.156 per mouse click and 0.024 per second for WCSPP. The approximation performance decreased by 0.089 per mouse click and 0.01 per second for ETSP, and by 0.101 per mouse click and 0.011 per second for RTSP (all results are significant at $p < 0.001$). Productivity was higher for ETSP than for RTSP (significant at $p < 0.001$ for approximate performance per click and at $p < 0.05$ for approximate performance per second).

B.4. *Random selection algorithms*

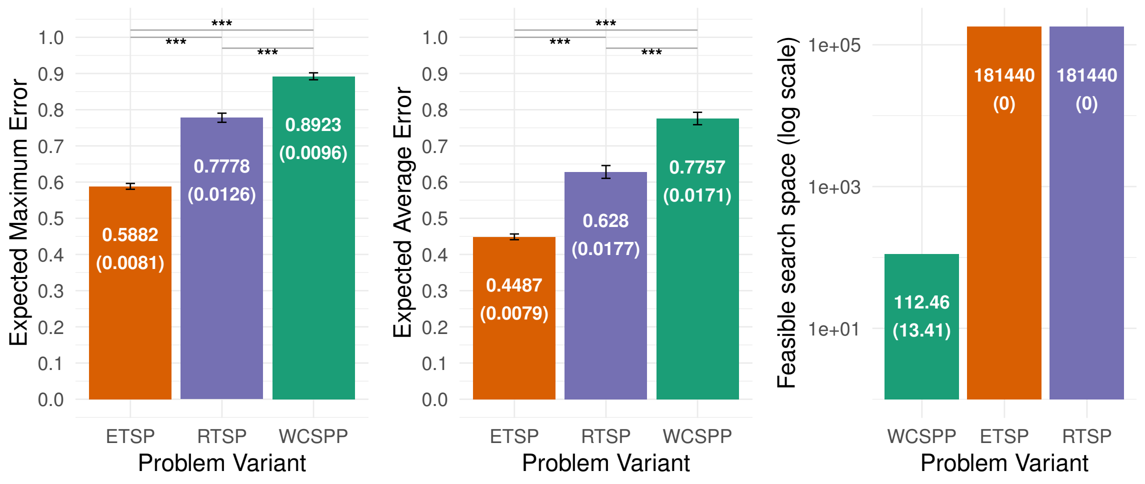
While approximation complexity classes are based on worst-case analysis, we studied average performance on random instances of three classes. Because the identity of the worst

TABLE B.VI

MIXED EFFECTS MODEL FOR PRODUCTIVITY WITH PARTICIPANTS AS RANDOM INTERCEPT.

	Dependent variable:	
	Aproximation performance/Clicks	Aproximation performance/Time
	(1)	(2)
Fixed effects [Estimate (Std. Error)]		
Intercept (WCSPP)	0.156^{***}	0.024^{***}
	(0.004)	(0.001)
ETSP	-0.089^{***}	-0.010^{***}
	(0.004)	(0.001)
RTSP	-0.101^{***}	-0.011^{***}
	(0.004)	(0.001)
Random effects (Participant, $n = 24$)		
σ^2	0.0035	0.0001
ICC	0.0842	0.1384
Observations	1,624	1,624
Marginal R^2 / Conditional R^2	0.348 / 0.403	0.145 / 0.264
Note:		$^{*}p<0.05$; $^{**}p<0.01$; $^{***}p<0.001$

case depends on the algorithm used, and humans exhibit enormous diversity in solution strategies (across instances and individuals), we set out to determine whether a random-selection strategy would generate the effect we observe among humans.



*** denotes p-value less than 0.001; Values in brackets denote standard error.

FIGURE B.5.—Measure of error bounds and search space size for all problem variants.

We investigate whether the errors from randomly selecting potential solutions differs across problem variants. To do this, we first construct the feasible solution space for the instances. For TSP the feasible solution space includes all cyclic paths that visit each city once. For WCSPP the feasible solution space includes all paths that begin at the start city, terminate at the end city, and whose total cost is at most the given threshold. Each entry in the feasible solution space represents a feasible solution. An optimal solution is a feasible solution with the minimum value. The quality of a feasible solution can be measured in terms of how far it is from the optimal, or in other words, how much error a decision maker would have made if they submitted that particular feasible solution. An error for a feasible solution w is defined as $1 - \mathsf{value}(w^*)/\mathsf{value}(w)$ where w^* is an optimal solution. We investigate the distribution of these errors. We compute two metrics for the errors: the expected maximum error, computed by averaging the maximum possible error (that is, the error of the feasible solution whose value is farthest from the optimal value), and the expected average error, computed by first averaging errors for an instance across feasible solutions, and then taking the mean of all the averages.

Expected errors are comparable only between ETSP and RTSP. This is because the underlying problem definition is the same only for these two variants. For example, for ETSP and RTSP, all feasible solutions have the same length and the size of the feasible solution set is also the same. In contrast, for WCSPP the size of the feasible solution set is significantly lower than that for ETSP and RTSP (see Figure B.5). In fact for WCSPP the size of the feasible solution set will also tend to increase with an increase in the cost threshold. Hence, the size of the feasible solution set will vary across different instances of WCSPP. In addition, for WCSPP, optimal solutions may be of shorter length (in terms of the path size) as compared to the length of other feasible solutions. Hence, given that most algorithms incrementally construct feasible solutions, some of the feasible solutions may not be pragmatic to consider.

We compare ETSP and RTSP in Figure B.5 (results for WCSPP are provided *pro memoria*). The expected maximum error and the expected average error are higher for RTSP as compared to ETSP (and significant at $p < 0.001$). Suppose one had a very inefficient algorithm (one that has a very high error). Such an algorithm will have an expected maximum error of 0.7778 for RTSP and 0.5882 for ETSP. Similarly, if one had to take a guess on the performance of an unknown algorithm, the uninformed prior would be the expected average error, which is higher for RTSP (0.628) than ETSP (0.4487). Hence, the expected errors are guided by the structural properties of the instances.

REFERENCES

ARORA, SANJEEV AND BOAZ BARAK (2009): *Computational complexity: a modern approach*, Cambridge University Press. []

ARTHUR, W BRIAN (2018): *The economy as an evolving complex system II*, CRC Press. []

AUSIELLO, GIORGIO, PIERLUIGI CRESCENZI, GIORGIO GAMBOSI, VIGGO KANN, ALBERTO MARCHETTI-SPACCAMELA, AND MARCO PROTASI (2012): *Complexity and approximation: Combinatorial optimization problems and their approximability properties*, Springer Science & Business Media. []

BEŞLIU, CORINA (2022): "Complexity in insurance selection: Cross-classified multilevel analysis of experimental data," *Journal of Behavioral and Experimental Finance*, 100713. []

BHARGAVA, SAURABH, GEORGE LOEWENSTEIN, AND JUSTIN SYDNOR (2017): "Choose to lose: Health plan choices from a menu with dominated option," *The Quarterly Journal of Economics*, 132, 1319–1372. []

BOSSAERTS, PETER, ELIZABETH BOWMAN, FELIX FATTINGER, SHIJIE HUANG, CARSTEN MURAWSKI, SHIREEN TANG, AND NITIN YADAV (2020): "Asset Pricing under Computational Complexity," Tech. rep., SSRN Working Paper 3475433. []

BOSSAERTS, P AND C MURAWSKI (2017): "Computational complexity and human decision-making," *Trends in Cognitive Science*, 21, 917–929. []

CHALMERS, DAVID J (2011): "A computational foundation for the study of cognition," *Journal of Cognitive Science*, 12, 325–359. []

COLLIARD, JEAN-EDOUARD AND CO-PIERRE GEORG (2020): "Measuring regulatory complexity," . []

CRAWFORD, IAN AND BRAM DE ROCK (2014): "Empirical revealed preference," *Annu. Rev. Econ.*, 6, 503–524. []

FRANCO, JUAN PABLO, KARLO DOROC, NITIN YADAV, PETER BOSSAERTS, AND CARSTEN MURAWSKI (2021a): "Task-independent metrics of computational hardness predict performance of human problem-solving," *bioRxiv*. []

FRANCO, JUAN PABLO, NITIN YADAV, PETER BOSSAERTS, AND CARSTEN MURAWSKI (2021b): "Generic properties of a computational task predict human effort and performance," *Journal of Mathematical Psychology*, 104, 102592. []

FRIXIONE, MARCELLO (2001): "Tractable competence," *Minds and Machines*, 11, 379–397. []

GAREY, MICHAEL R AND DAVID S JOHNSON (1979): *Computers and intractability*, vol. 174, freeman San Francisco. []

GIGERENZER, GERD AND REINHARD SELTEN (2002): *Bounded rationality: The adaptive toolbox*, MIT press. []

GIGERENZER, GERD AND PETER M TODD (1999): "Fast and frugal heuristics: The adaptive toolbox," in *Simple heuristics that make us smart*, Oxford University Press, 3–34. []

GILBOA, ITZHAK, ANDREW POSTLEWAITE, AND DAVID SCHMEIDLER (2021): "The complexity of the consumer problem," *Research in Economics*, 75, 96–103. []

KAKADE, SHAM MACHANDRANATH (2003): *On the sample complexity of reinforcement learning*, University of London, University College London (United Kingdom). []

KELLERER, HANS, ULRICH PFERSCHY, AND DAVID PISINGER (2004): "Introduction to NP-Completeness of knapsack problems," in *Knapsack problems*, Springer, 483–493. []

KOLMOGOROV, ANDREI N (1983): "Combinatorial foundations of information theory and the calculus of probabilities," *Russian Mathematical Surveys*, 38, 29–40. []

MACGREGOR, JAMES N AND YUN CHU (2011): "Human performance on the traveling salesman and related problems: A review," *The Journal of Problem Solving*, 3, 2. []

MELOSO, DEBRAH, JERNEJ COPIC, AND PETER BOSSAERTS (2009): "Promoting intellectual discovery: patents versus markets," *Science*, 323, 1335–1339. []

MOORE, CRISTOPHER AND STEPHAN MERTENS (2011): *The nature of computation*, OUP Oxford. []

MURAWSKI, CARSTEN AND PETER BOSSAERTS (2016): "How humans solve complex problems: The case of the Knapsack problem," *Scientific reports*, 6, 34851. []

NEDELEC, THOMAS, CLÉMENT CALAUZÈNES, NOUREDDINE EL KAROUI, VIANNEY PERCHET, ET AL. (2022): "Learning in repeated auctions," *Foundations and Trends® in Machine Learning*, 15, 176–334. []

NEWELL, ALLEN, HERBERT ALEXANDER SIMON, ET AL. (1972): *Human problem solving*, vol. 104, Prentice-hall Englewood Cliffs, NJ. []

OPREA, RYAN (2020): "What makes a rule complex?" *American economic review*, 110, 3913–51. []

OTWOROWSKA, MARIA, MARK BLOKPOEL, MARIEKE SWEERS, TODD WAREHAM, AND IRIS VAN ROOIJ (2018): "Demons of Ecological Rationality." *Cogn. Sci.*, 42, 1057–1066. []

PAYZAN-LENESTOUR, ELISE AND PETER BOSSAERTS (2015): "Learning about unstable, publicly unobservable payoffs," *The Review of Financial Studies*, 28, 1874–1913. []

PELÁNEK, RADEK (2011): "Difficulty rating of sudoku puzzles by a computational model," in *Twenty-Fourth International FLAIRS Conference*. []

ROUGHGARDEN, TIM AND INBAL TALGAM-COHEN (2019): "Approximately optimal mechanism design," *Annual Review of Economics*, 11, 355–381. []

RUIZ-VANOYE, JORGE A., JOAQUÍN PÉREZ-ORTEGA, RODOLFO A. PAZOS R., OCOTLÁN DÍAZ-PARRA, JUAN FRAUSTO-SOLÍS, HECTOR J. FRAIRE HUACUJA, LAURA CRUZ-REYES, AND JOSÉ A. MARTÍNEZ F. (2011): "Survey of polynomial transformations between NP-complete problems," *Journal of Computational and Applied Mathematics*, 235, 4851–4865, congressional Contributions to Computational and Applied Mathematics: ICCAM2009. []

SAMUELSON, PAUL A (1948): *Fundamentals of Economic Analysis*, Cambridge, MA: Harvard University Press. []

SCHAEFER, THOMAS J (1978): "The complexity of satisfiability problems," in *Proceedings of the tenth annual ACM symposium on Theory of computing*, 216–226. []

SIMON, HERBERT A (1979): "Rational decision making in business organizations," *The American economic review*, 69, 493–513. []

TVERSKY, AMOS AND DANIEL KAHNEMAN (1974): "Judgment under uncertainty: Heuristics and biases," *science*, 185, 1124–1131. []

VAN ROOIJ, IRIS, MARK BLOKPOEL, JOHAN KWISTHOUT, AND TODD WAREHAM (2019): *Cognition and Intractability: A Guide to Classical and Parameterized Complexity Analysis*, Cambridge University Press. []

VAN ROOIJ, IRIS, ALISSA SCHACTMAN, HELENA KADLEC, AND ULRIKE STEGE (2006): "Perceptual or Analytical processing? Evidence from children's and adult's performance on the Euclidean traveling salesperson problem," *The Journal of Problem Solving*, 1, 6. []

VAN ROOIJ, IRIS AND TODD WAREHAM (2012): "Intractability and approximation of optimization theories of cognition," *Journal of Mathematical Psychology*, 56, 232–247. []

VEBLEN, THORSTEIN (1908): "Professor Clark's Economics," *The Quarterly Journal of Economics*, 22, 147–195. []

WEGENER, INGO (2005): *Complexity theory: exploring the limits of efficient algorithms*, Springer Science & Business Media. []

Co-editor [Name Surname; will be inserted later] handled this manuscript.