8

Complex Specified Information (CSI) Collecting

Eric Holloway

Independent Scholar

Abstract

Intelligent Design Theory makes use of the concept of intelligent agency as a distinct causal mode, distinct from chance or algorithmic modes of causation. The influence of such agency is often detected by the contribution of information to a search process. Assuming humans are capable of such causal roles, then it should be possible to measure the amount of information that a human is contributing to such a process. This is done by measuring the success rate for a search for a solution to a computationally hard problem by both humans and computers. The methodology used for this experiment was not successful, but it is hoped that the experimental setup and methodology will inspire further improvement and research in this area.

1 Introduction

Research comparing human cognitive capabilities to computer algorithms suggest humans possess supra-computational cognition. Humans are capable of finding good solutions to computationally intractable problems, and this capability scales at a faster rate with larger problems than the best known algorithms (Dry, Lee, Vickers, & Hughes, 2006). Many popular games are algorithmically intractable to solve (Viglietta, 2011). Programmers appear capable of solving halting problems (Bartlett, 2014a). People can solve insight problems, for which there is currently no known computational method (Bartlett, 2014b). Observers can pick out targets in a picture relevant to their goal independently of the target's features (Gue, Preston, Das, Giesbrecht, & Eckstein, 2012). Such capabilities either have never been algorithmically

emulated, or violate well known and well substantiated computational constraints, such as the computational complexity of the problem (Cormen, Leiserson, Rivest, & Stein, 2001), the No Free Lunch Theorem (NFLT) (Wolpert & Macready, 1996, 1995) and the halting problem (Cover & Thomas, 2006).

Intelligent Design Theory (IDT) provides a formal language to account for observations of supra-computational cognition. According to IDT, intelligent agents, such as humans, are capable of creating a new form of information called complex specified information (CSI) (Dembski, 2005). Dembski's work on search algorithms (Dembski, 2006) implies that if human interaction can be incorporated into a search and optimization algorithm, these algorithms can surpass the limitations of the NFLT over a wide variety of hard problems. This is possible due to the unique ability of intelligent agents, such as humans, to create CSI. A further implication is the user does not need specialized interfaces or training for each type of problem. The possibility of such a generalized interface is suggested by research demonstrating context independent problem solving by humans (Sperber, Cara, & Girotto, 1995) and suggested by IDT's implications.

2 Problem Description

In the computational domain, it is problematic to identify whether human interactions can be defined by an algorithm. Since all the interactions with the computer are defined by a series of 1s and 0s, i.e., a bit string, the interaction can be codified by an algorithm that outputs the same bit string. Consequently, on first analysis, trying to computationally distinguish human interaction from algorithm output seems impossible. If human interaction is indistinguishable from algorithmic output, then it is not plausible human interaction can surpass the No Free Lunch Theorem (NFLT). Such is the basic problem addressed: How is it possible to distinguish human interaction from algorithm output?

Even though it is always possible to codify a series of events in a finite domain after the fact, the question is whether such codification is possible prior to the event's occurrence. A prior codification is not possible in all cases; otherwise, it violates the halting problem (Cover & Thomas, 2006). This means the possibility of codifying a human's interaction after the interaction has occurred does not necessarily entail it can be codified before the interaction. Consequently, it is possible that with the right experimental design a human's interaction can be distinguished from an algorithm output.

3 Background

The question is, what is the experimental design that can make the distinction between human and algorithm? One approach is to use the NFLT. The theorem sets

precise, rigorous boundaries on the mathematical capabilities of algorithms. Anything performing outside of these boundaries is by logical necessity non-algorithmic. To show humans are supra-computational, one need simply show they do not abide by the NFLT. Unfortunately, such a demonstration may be impossible. The NFLT only applies across all problems within a very small, specialized subset of problem types. As such, the NFLT is quite difficult to apply in practice.

The Almost No Free Lunch Theorem (ANFLT) (Droste, Jensen, & Wegener, 2002) provides a solution. The theorem shows while the original NFLT does not apply to most particular problem domains, the expected performance for most algorithms on many real world problems does not vary significantly between algorithms. There is almost no free lunch for a large portion of real world problems.

The ANFLT result implies that given a search algorithm, as long as the algorithm is selected independently of the problem, it is unlikely the choice of algorithm makes a significant difference in search performance. Consequently, if human interaction discovers a solution significantly better than search algorithms, it is very likely the interaction was non-algorithmic, and therefore supra-computational.[1]

According to William Dembski's research on search algorithms (Dembski & Marks II, 2010), this supra-computational interaction takes the form of active information creation. Active information is the information necessary to improve a search algorithm's expected speed in finding a target solution beyond that of a random search. That is, with more active information, fewer search queries are necessary to find the target solution.

Active information can be inserted into the algorithm from an existing source of external information, or it may be created. In the case of the ANFLT, if the search algorithm exhibits significant amounts of active information (i.e., finds a target much faster than statistically expected), this information must be created since the conditions of the ANFLT prohibit the information from being merely transferred from an external source.

Since the information is created, it cannot come from either chance or necessity, as these sources can only degrade or transfer already existing information. Additionally, the information is specified by the degree it reduces the number of search queries to find the target solution. Information that is neither the product of chance nor necessity and is specified is complex specified information (CSI) (Dembski, 2005) as defined by Intelligent Design Theory (IDT). Furthermore, IDT claims CSI can come from intelligent agents.

[1]Throughout this paper, the term "solution" refers to both the best/true solution to a problem, as well as substandard solutions to a problem. Even in the case where there is only one true answer to a problem, there may be other answers that are approximations of the true answer. The two types of solutions are distinguished in terms of their optimality. The true solution is the optimal solution.

4 Approach

The approach in this study is to develop and test a general technique for integrating human interaction into search and optimization algorithms. By incorporating human interaction in this way, it is possible to determine whether humans violate the algorithmic search constraints of the ANFLT and consequently create CSI as predicted by IDT. This technique's implementation consists of a software framework that can be integrated with many different kinds of algorithms and problems, including both single and multi-objective problems. The technique is referred to as CSI Collecting (CSIC) throughout the rest of this paper.

CSIC is demonstrated as a proof of concept by using it to find public and private keys for the RSA asymmetric cryptography system (Cormen et al., 2001). The algorithm used in CSIC is a multi-objective genetic algorithm (Coello, Lamont, & Veldhuizen, 2007). The human users will use CSIC through Amazon's Mechanical Turk service.

Success in the experiment is measured by the rate of solution improvement (fitness increase) compared with problem information discovered, which is Fitness/Information or FI for short. Solution improvement is measured by an objective function in the genetic algorithm. The amount of problem information discovered is measured by the number of solutions evaluated. The improvement due to human interaction is compared to the improvement due to the genetic algorithm as measured by FI.

Since the project is currently in the exploratory stage, the comparison is informal. It is assumed the algorithm discovers problem domain information at an exponentially greater rate compared to human agents. Consequently, any greater or equivalent improvement of fitness by the human agents compared to the algorithm produces a very high FI value in favor of the human agents. Thus, the comparison between human agents and the genetic algorithm is based purely on fitness increase.

5 Implementation

The general technique used is as follows. A standard multi-objective genetic algorithm is used, a type of stochastic global search algorithm (Figure 8.1). A genetic algorithm takes a set of solutions, measures how good each solution is according to some fitness valuation function, varies the solutions to generate a new set using variation operators such as mutation and crossover, and then from both sets of solutions selects an output set according to some criteria. The algorithm then reiterates this process on each subsequent output set until a stopping criteria is reached. The solutions themselves are represented to the algorithm as fixed length bit strings. The functions *generate* and *select* in Figure 8.1 can each incorporate an intelligent agent. In the implemented version of CSIC, only the *select* function incorporates human interaction, even though the more idealized version can incorporate human interaction in the *generate* phase

$\mathcal{S}_i^p \leftarrow \{init()\}$;
$\mathcal{S}_o^f[0] \leftarrow \{\}$;
$t := 0$;
Ensure: $\forall s^p \{s^p \in \mathcal{S}[t] \rightarrow s^p \in \mathcal{S}_i^p\}$
Ensure: $\forall s^f \{s^f \in \mathcal{S}^f[t] \rightarrow s^f \in \mathcal{S}[t] \wedge \mathcal{F}(s^f)\}$
Ensure: $\forall s^f \{s^f \in \mathcal{S}_o^f[t] \rightarrow s^f \in \mathcal{S}^f[t] \wedge \mathcal{C}(s^f, \mathcal{S}_o^f[t-1])\}$
while $\mathcal{O}(\mathcal{S}_o^f[t], t) \neq TRUE$ **do**
$t := t + 1$;
$\mathcal{S}[t] := generate(\mathcal{S}_i^p, \mathcal{S}[t-1])$
$\mathcal{S}^f[t] := feasible(\mathcal{S}[t])$
$\mathcal{S}_o^f[t] := select(\mathcal{S}^f[t], \mathcal{S}_o^f[t-1])$
end while

p	partial solution
f	feasible solution
i	input set
o	output set
$\mathcal{S}$	set of solutions, can be either partial or feasible
$\mathcal{F}$	feasibility, function of solution feasibility, returning true or false
$\mathcal{C}$	comparison, whether solution is selected when compared to solution set
$\mathcal{O}$	objective, function of both output solution set and current iteration count

Figure 8.1: Stochastic global solution search algorithm

of the algorithmic process.

5.1 Data Flow

The data flow between the human and the algorithm is demonstrated in Figure 8.2. Both the unguided and guided algorithms are run in tandem on the same solution set for greater effectiveness in discovering new solutions. For experimental purposes, this does muddy the data to an extent, but not irrevocably. Human-provided solutions are uniquely identified, and the phylogeny of each solution is tracked, making it possible to derive the impact of human interaction.

The two processes are combined into a hybrid system because for real world use algorithms and humans work well together. The algorithmic side is very good at checking many solutions very quickly, thereby providing the user with better information for making decisions on new areas of the problem to explore.

5.2 User Interface

In the user interface, the user is presented with a list of solutions from which he can make his selection. The solutions are the strings of arbitrary symbols in Figure 8.3.

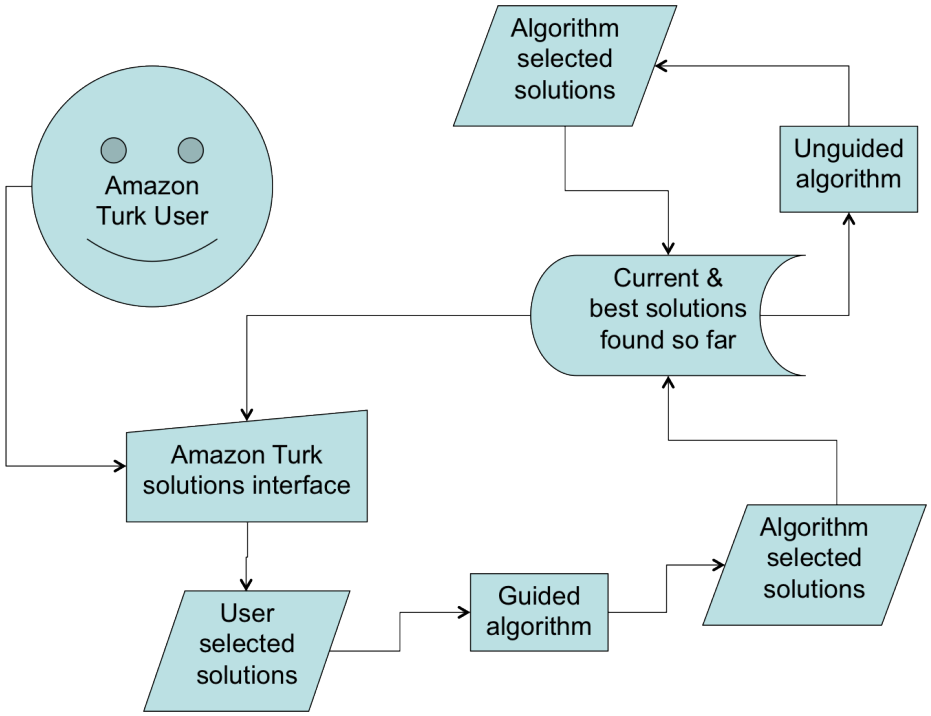

Figure 8.2: CSIC data flow between user and algorithm.

Each solution is represented symbolically, so the user has no contextual clues as to what the problem is that he is attempting to solve. This is done in order to create a context independent interface that can be used for a wide variety of problems and algorithms. The only information to which the user has access is similarities between solutions (as shown by similarities in symbols) and the value of each solution. Once the user selects a set of solutions, he picks the type of variation operators to apply to the solutions and then presses "GO" to have the algorithm, as shown in Figure 8.1, create a new set of solutions. Dembski's work implies that even with only this information it is possible for the user to be able to provide active information to the search algorithm because a prior external source of information is unnecessary to add active information to the search since the user is an intelligent agent.

The types of operators have different credit costs since some operators discover more information about the problem than others. It is important to track the amount of information with which the user is making his decisions in order to fairly compare his performance to that of the unguided algorithm. The user is rewarded based on whether this new set of solutions improves over the best solutions found so far.

The actual implementation in Figure 8.4 is a simplified version of the interface shown in Figure 8.3, since it is used by people on the Amazon Turk service. Due to the low wage for using the interface, they cannot be expected to spend a long time trying to understand the intricacies of different variation operators. Accordingly, credits are done away with since the same operators are used in every iteration. Additionally, the numerical score is replaced with a visual indication of solution value, where higher

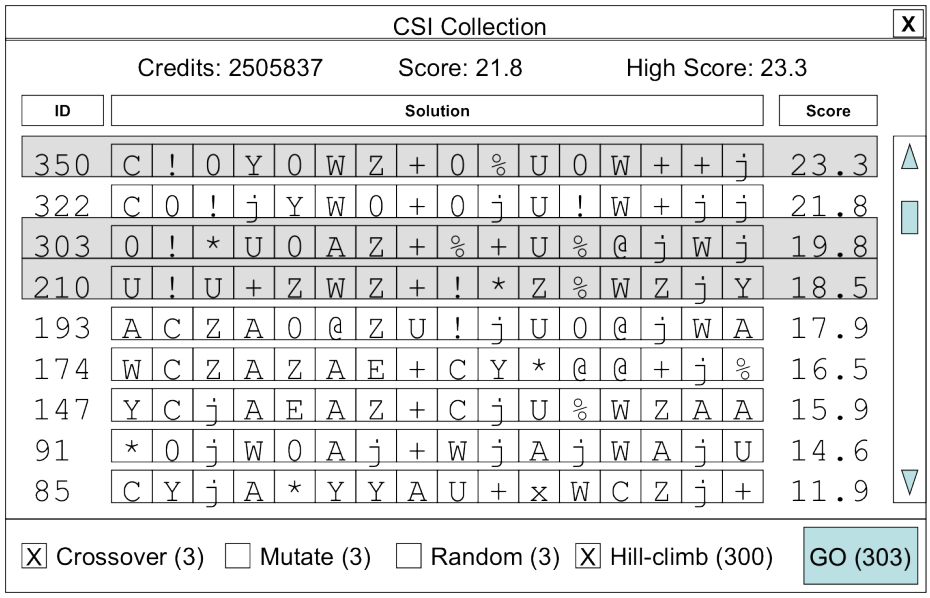

Figure 8.3: User interface concept for CSIC.

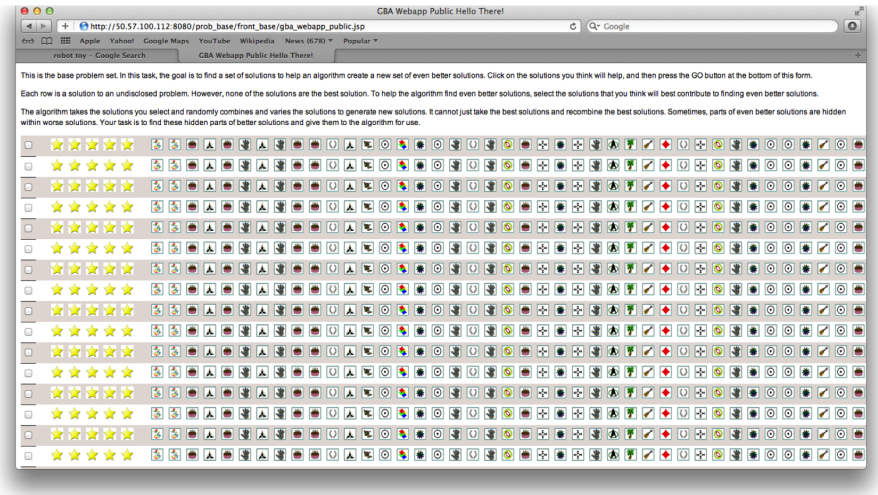

Figure 8.4: Actual user interface website.

valued solutions have a greater number of stars.

Otherwise, the basic idea is the same between the actual and conceptual interfaces. The users select solutions to guide the search algorithm and then press "GO." The users are paid a basic wage for using the webpage and are rewarded with a bonus based on the value of new solutions discovered. The intent is to financially incentivize the users to discover highly valued and highly unique solutions.

5.3 Problem

CSIC is used to find the prime factors that generate keys for an asymmetric cryptosystem. There is no known correlation between finding prime factors and multi-objective genetic algorithms, nor with human input during the selection phase. This means the problem and algorithm together meet the criteria for the ANFLT to apply.

Breaking asymmetric encryption is also an extremely hard problem, which is why it forms the basis of much information transmission security. The asymmetric cryptosystem is RSA (Cormen et al., 2001). The genetic algorithm uses a multi-objective fitness function to measure how close a solution is to the correct set of primes. To find the primes, the fitness function is given access to a true plaintext and true cyphertext. The two objectives are:

1. *Matching encrypted bits.* The plaintext is encrypted using the keys generated by the solution primes. This encrypted text is then compared with the true cyphertext, and the number of matching 1 bits are counted. The 0 bits are not counted because they generally overwhelm the number of 1 bits.

2. *Matching decrypted bits.* The true cyphertext is decrypted using the solution keys, and its accuracy is measured as for the previous objective.

6 Results and Conclusion

The guided genetic algorithm received 500 human inputs from the Amazon Mechanical Turk service. The human input contributed 1 out of 18 superior solutions found (Figure 8.5). The one solution found by a human was also found by the algorithm. However, the logs show all human input consisted of selecting every single solution, which can be replicated by an algorithm.

This result did not validate the hypothesis, though the failure is due to methodological factors, as will be discussed in Section 8. Since both the human and algorithm discovered the same solution, it is not clear the solution was originally found by the human. Thus, in terms of the FI metric in Section 4, it is not possible to say whether the human outperformed the algorithm.

7 Theoretical Objections

While the methodology for CSIC is still in its infancy, and its efficacy has yet to be demonstrated, there are also theoretical objections to the concept that have been or could be raised. The following section attempts to provide answers to these objections. In all of these objections the assumption is made that CSI, or active information, cannot be created by algorithmic sources. Additionally, it is also assumed that active information is a subcategory of CSI.

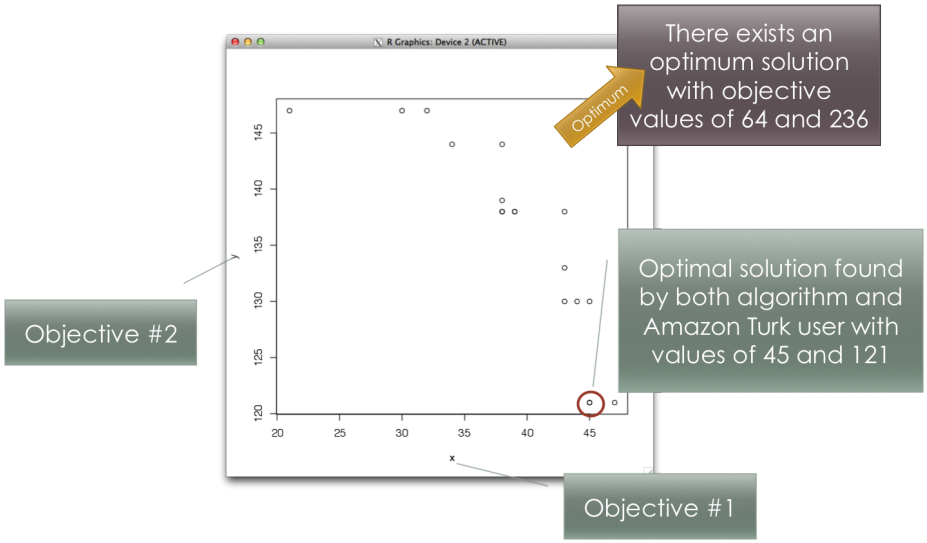

Figure 8.5: Best solutions found from Amazon Turk experiment. The objectives are described in Section 5.3.

7.1 Objection: Human intelligence is an algorithm with already high amounts of CSI

One objection to this methodology is that human intelligence is an algorithm that *already contains* high amounts of CSI, especially since the ANFLT does not rule out the possibility that a proximate algorithmic information source is available. It might be the case that human minds are highly tuned algorithms for particular difficult problem domains. Consequently, humans will outperform all current state of the art algorithms in these domains, but this does not preclude human intelligence being algorithmic. This objection holds even in the case of the ANFLT.

Such an objection is valid. The CSIC experiment cannot categorically rule out the possibility that human intelligence is a kind of algorithm. However, CSIC can provide an inference to the best explanation between two competing hypotheses. The first hypothesis asserts that the proximate cause, human intelligence, created the information, while the second hypothesis states that a more remote intelligent agent created the information.

To determine which hypothesis is better supported by a positive result in the CSIC experiment, the principle of minimal CSI creation is proposed. The principle is analogous to Ockham's Razor. This principle states in the case where the creation of CSI is presupposed, the explanation relying on the least amount of CSI creation is preferred.

The first hypothesis relies on less CSI creation since the human only creates CSI for a particular problem instance. Hypothesis two requires the creation of enormous amounts of CSI covering all possible problem instances the human might encounter.

Consequently, while the CSIC experiment does not rule out algorithmic human intelligence, it does show that supra-computational human intelligence is the best explanation of a positive result.

7.2 Objection: Supra-computation does not imply effectiveness on all problems

Another objection to this methodology is that even if human intelligence is supra-computational, this does not imply that it is necessarily effective on all possible problems. It could be performing supra-computational actions which are still limited in scope and thus preventing its use in arbitrary problems. This objection is also valid. Consequently, the question is, in which problems do humans demonstrate supra-computational abilities? One way to answer this question is experimentally, by having humans work with different problems and seeing in which instances supra-computational abilities are exhibited. This is the intent of the CSIC experiment. There is no presumption supra-computational abilities will be demonstrated, only a presumption that it is possible to detect such abilities within the experiment.

Research by Bartlett (2014b) suggests that even if humans do demonstrate supra-computation, such a capability may depend on existing information, such as axioms about the problem domain. If true, then the almost complete removal of context in CSIC may render the user incapable of contributing active information to the search algorithm.

7.3 Objection: Supra-computation is only exhibited in context dependent interactions

This objection states that the removal of almost all context in CSIC is a methodological problem because experience seems to indicate that supra-computation only occurs within situational contexts. A solution can only be created when the problem is contextually understood.

However, the observation is not always true. One scenario similar to CSIC where information is created is learning to read. When people learn to read a language, they are presented with a string of symbols without an inherent context. They learn the meaning of the symbols through external responses, such as getting affirmation when mapping the symbols to the correct action or object.

7.4 Objection: Supra-computation requires holistic reasoning

This objection states that in order to solve a problem, the person solving must not only know the problem, but also understand why the problem exists. Since the user

of CSIC neither knows nor understands the problem at hand, he will be unable to solve the problem.

There is a degree of holism available in CSIC, as the user can see many solutions and valuations and thus look for overarching patterns. The user can understand at a very abstract level the characteristics of good and bad solutions.

7.5 Objection: Fitness function in experiment cannot factor primes for RSA keys

This objection is quite likely correct. However, even if correct, if human interaction improves the solutions beyond the algorithmic limits, then the experiment achieves a positive result. On the other hand, better choices of problems, such as well understood pedagogical problems, would greatly improve the experimental design.

RSA cracking was chosen for this experimental investigation mostly because the applicability of solving this problem is much easier to explain to a lay audience than more pedagogical problems. Additionally, a successful result for this problem would have direct, groundbreaking relevance for software engineering.

8 Future Work

While the essential methodology and implementation of CSIC have now been created and tested, many areas of improvement remain. Additionally CSIC must be compared to alternatives to see if CSIC is truly effective and beneficial.

The main potential area of improvement is in verifying whether the Amazon Turk input is truly human generated and whether the users are actually trying to find patterns. Users have been known to script Amazon Turk jobs, and without such verification in this case it is not possible to know with certainty whether the input is human generated. When the logs from this experiment were analyzed, it turns out the Amazon Turk users were just selecting all the solutions and not trying to find patterns in the solutions. To provide a cleaner environment for experimentation, the guided and unguided algorithms should be separated. Additionally, the data logging needs to be time-stamped.

The algorithm and problem used in the experiment can be improved in numerous ways. Pedagogical problems and algorithms should be used to compare the effectiveness of CSIC to the current state of the art. Practical problems where search algorithms have already proven effective should be explored to see if CSIC can provide additional benefit. CSIC should also be compared to contextualized human-powered search algorithms, such as Foldit, to see how the addition of context affects search effectiveness.

Different forms of motivation should be explored. The Amazon Turk interface relies on a financial motivation, which motivates users to cut corners and perform

the task in the fastest way possible. Such motivation does not encourage users to find extremely good solutions. If the interface is in the form of an entertaining game, users are better encouraged to find good solutions. Furthermore, a simplified explanation of the cutting edge relevance of their work provides an intrinsic motivation, and encourages innovation as demonstrated by significant user innovation in Foldit (Moore, 2012).

References

Bartlett, J. (2014a). Calculating software complexity using the halting problem. In J. Bartlett, D. Halsmer, & M. R. Hall (Eds.), *Engineering and the ultimate* (pp. 123–130). Broken Arrow, OK: Blyth Institute Press.

Bartlett, J. (2014b). Using Turing oracles in cognitive models of problem-solving. In J. Bartlett, D. Halsmer, & M. R. Hall (Eds.), *Engineering and the ultimate* (pp. 99–122). Broken Arrow, OK: Blyth Institute Press.

Coello, C. A. C., Lamont, G. B., & Veldhuizen, D. A. V. (2007). *Evolutionary algorithms for solving multi-objective problems*, chapter MOEA Parallelization. Springer: New York.

Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2001). *Introduction to algorithms*. MIT Press, second edition.

Cover, T. M. & Thomas, J. A. (2006). *Elements of information theory*. Wiley-Interscience, 2nd edition.

Dembski, W. (2005). Specification: The pattern that specifies intelligence. Available from http://www.designinference.com/documents/2005.06.Specification.pdf

Dembski, W. (2006). *Conservation of information in search*. Technical report, Center for Informatics.

Dembski, W. A. & Marks II, R. J. (2010). The search for a search: Measuring the information cost of higher level search. *Journal of Advanced Computational Intelligence and Intelligent Informatics*, 14(5), 475–486.

Droste, S., Jansen, T., & Wegener, I. (2002). Optimization with randomized search heuristics: the (a)nfl theorem, realistic scenarios, and difficult functions. *Theoretical Computer Science*, 287, 131–144.

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

Gue, F., Preston, T. J., Das, K., Giesbrecht, B., & Eckstein, M. P. (2012). Feature-independent neural coding of target detection during search of natural scenes. *The Journal of Neuroscience*, 32(28), 9499–9510.

Moore, E. A. (2012). Foldit game leads to aids research breakthrough. Available from http://news.cnet.com/8301-27083_3-20108365-247/foldit-game-leads-to-aids-research-breakthrough/

Sperber, D., Cara, F., & Girotto, V. (1995). Relevance theory explains the selection task. *Cognition*, 57, 31–95.

Viglietta, G. (2011). Gaming is a hard job, but someone has to do it! *arXiv*. Available from http://arxiv.org/pdf/1201.4995.pdf

Wolpert, D. H. & Macready, W. G. (1995). *No free lunch theorems for search*. Technical report, Santa Fe Institute.

Wolpert, D. H. & Macready, W. G. (1996). *No free lunch theorems for optimization*. Technical report, Santa Fe Institute and TXN Inc.