

Social Computing Article*

Abstract

Social collaboration has been shown to facilitate problem-solving activity in a diverse set of environments. Nevertheless, if not well designed, a social human computation system may achieve results as poor as those of a single human subject performing a task. This scenario reflects a need for a better understanding of the performance issues of human problem-solving social networks. We present a novel method for artificial social problem solving and test it against results with human subjects, analysing which parameters have effects on the performance of problem-solving social networks. Results show that evaluating neighbours' solutions in order to decide from which neighbour to copy is significantly better than selecting solutions based on their position on the graphic interface, as tests with human subjects have shown to be the preferred behaviour among people. Our results also indicate that copying and guessing are highly beneficial to the performance of these social networks. We then use these results to propose a collection of guidelines concerning the design of an efficient human computation system.

Introduction

For years multi-agent systems have been used to research cooperation as a tool for problem-solving. Recently, however, there has been an increasing interest in the study of human beings as problem-solving agents. Several experiments have been conducted in which subjects are connected in a network with the goal of collectively solving a specific problem, and those have helped shed some light on the way humans interact to solve problems. However, although those studies can provide us with observations and hypotheses, there is still a difficulty in finding ways to explain the observed behaviour. In this paper, we use a multi-agent based simulation to complement the study of human computation, as a way of explaining the strategies used by humans and understanding their consequences in a cooperative environment. The control that simulations provide us over the behaviour of the agents allows us to better understand possible reasons for the ones displayed by humans.

Human beings are known to be able to easily perform tasks which are still generally difficult for computers, such

as natural language processing and image recognition. However, it would be useful to be able to apply human social computation to more straightforward computer problems with precise definitions and algorithms, but which are still computationally intensive. The different sets of abilities between computers and humans suggest that the latter might provide a new approach to those problems that might be more effective than current techniques employed by computers. That notion has been successfully applied, for example, in one of the most significant problems in the field of bioinformatics: protein structure prediction (PSP). A software that presents the problem to humans as an online computer game, called *Foldit* (Cooper 2011), has produced significant results in the research of PSP, which is usually approached as an optimization problem requiring extensive computational power. Those results have been attributed to human visual problem-solving and decision-making abilities, as well as social collaboration (Cooper 2011). However, we still don't know the limits of human abilities in problem-solving and how they compare to more traditional techniques. To take full advantage of human problem-solving strategies, we must learn their limitations. In order to obtain that knowledge, our system simulates human behaviour according to findings of social computation experiments.

Background

Social computation is a relatively new area of research with a diversified, interdisciplinary root that mixes social sciences, artificial intelligence, game theory and network science, among others. Only recently studies on the potential of human social networks for solving problems are gaining some popularity. Those have provided insights into, among other things, the impact of network structure in the collaboration process and the factors that lead neighbours' proposed solutions to be copied by individuals.

The origin of human computation as we know it today can be traced back to the work of (Ahn and Dabbish 2008), which identified the possibility of using entertainment as an incentive to participation of human subjects, applying it in games in which the participants are actually performing a computation. That is an idea that also appears in the *Foldit* game.

The series of experiments summarized in (Kearns 2012) are among the first to try to take advantage of the collec-

*This title is temporary

tive's problem-solving abilities to solve classical computer problems. The experiments are mostly based on the concept of coordination: subjects have individual incentives that are expected to drive them to cooperate with one another and lead them toward the collective goal.

Other initiatives have appeared, such as the ones by (Farenzena *et al.* 2011) and (Mason and Watts 2011), which take a different approach by having subjects trying to solve the collective problem individually, with the possibility of exchanging solutions between neighbours. Those have resulted in interesting conclusions on human behaviour when the possibility of copying peers makes itself present. It's in those models of experiments that we will focus in this work.

The experiments conducted by (Farenzena *et al.* 2011) had human beings trying to solve constraint satisfaction problems, namely Boolean Satisfiability (SAT) and the popular *Sudoku* game, with individuals connected through the network being able to exchange partial solutions of the problem in question. There were two main patterns of behaviour observed in this study in regard to copying neighbours' solutions:

- Humans might not evaluate the solutions proposed by their peers, instead choosing the most readily available one. On the other hand, (Mason and Watts 2011) suggest that their subjects did evaluate the available solutions, although an in-depth analysis of that wasn't provided.
- There is a higher probability of individuals copying peers' solutions when those are shared by several neighbours. That behaviour is referred to by the authors as *conformism*. A similar behaviour was observed by (Mason and Watts 2011), even though the experiments used a different problem.

(Mason and Watts 2011) also provided important insights to human cooperation. However, since our basis is the work of (Farenzena *et al.* 2011) we will use the former's results only as secondary support.

Contribution

We have developed and detailed a novel method for artificial social problem solving which draws inspiration from the *Memetic Networks* model proposed by (Araujo and Lamb 2008). It was used to build a multi-agent system that simulates the experiments of (Farenzena *et al.* 2011), using the model of human behaviour proposed by them. For simplicity, we limited our experiments to the *Sudoku* problem. Our goal is verifying whether the strategies employed by humans are competitive with other heuristics. After identifying the points in which those are lacking, we will then propose methods to increase the efficiency of human problem-solving networks.

System Overview

We developed a system in which the environment of the *Sudoku* problem-solving social network presented in (Farenzena *et al.* 2011) could be modelled through autonomous

agent networks and analysed through a series of experiments. We conceived a novel method for artificial social problem solving which draws inspiration from the optimization technique of *Memetic Networks* (Araujo and Lamb 2008).

A Novel Method for Artificial Social Problem Solving

A number of multi-agent methodologies that employ information flow through agents have been studied before. For instance, the *Memetic Networks* model proposed by (Araujo and Lamb 2008) draws inspiration from the phenomenon of *Cultural Evolution* discussed by Dawkins in (Dawkins 1976) and has a network of agents sharing, copying and incrementing units of information in a similar way that nature, according to Dawkins, deals with *Memes* (Dawkins 1976). Nevertheless, some aspects of social behaviour of great significance to social computing cannot be properly analysed through this methodologies. One example is the conformist behaviour studied in (Efferson *et al.* 2008) and (Farenzena *et al.* 2011). The proper study of these aspects demands a novel method for artificial collaborative problem solving.

We propose a method for solving computational problems by the means of a network of agents with social behaviour. Our algorithm is composed of an ordered set of N agents, each encoding a partial solution to the problem, and a binary $N \times N$ matrix representing possible connections between agents. Additionally, our algorithm is composed of two stages, namely the *Social Stage* and the *Cognitive Stage*.

- **Social Stage:** In the *Social Stage*, messages are passed from agent to agent through the network connections. Agents are thus presented with a multiplicity of messages. There is a particular probability associated with the behaviour of agents choosing to copy one of these solutions in contrast to keeping their current solutions. We call this probability the *Copy Rate*. When an agent chooses to copy, he or she is then supposed to select for copying only one of his or her received messages. This is done by the means of a particular message-selecting metric.
- **Cognitive Stage:** In the *Cognitive Stage*, agents are supposed to add local changes to the messages copied in the previous stage. This is done by the means of a heuristic or an exact method.

Performance Tests

In order to validate the algorithm as a social problem-solving technique, we have modelled the real-world collaborative *Sudoku* solving environment studied in (Farenzena *et al.* 2011) through our method and tested it over a set of *Sudoku* instances, having in mind that the emergence of similar results as those observed by (Farenzena *et al.* 2011) in their tests with *Sudoku* and *SAT* solving would validate our model as a good candidate for the design of social problem-solving simulations.

Initialize N agents, each encoding a partial solution to the problem;

```

while termination condition not met do
  for i = 1 to N do
    for j = 1 to N do
      if j is connected to i then
         $A_i = i_{th}$  agent;
         $A_j = j_{th}$  agent;
        Add  $A_j$ 's solution to the collection of
        messages of  $A_i$ ;
      end
    end
  end
  //Social Stage;
  for i = 1 to N do
     $A_i = i_{th}$  agent;
    selectedMessage = select( $A_i$ .messages);
    if random(0,1) < copyRate then
       $A_i$ .solution = selectedMessage;
    end
  end
  //Cognitive Stage;
  for i = 1 to N do
     $A_i = i_{th}$  agent;
    Add local changes to  $A_i$ 's solution
  end
end

```

Algorithm 1: No name yet

Social Stage

The experiments conducted by (Farenzena *et al.* 2011) point out a series of observations about the dynamics of cooperation in problem solving with human beings. For instance, the authors analysis has shown that human subjects are more likely to engage in the behaviour of copying the most readily available solutions on the graphic interface than in that of copying the most complete solutions. In order to analyse and compare this two behaviours, he have modelled each one of them as a message-selecting metric for the *Social Stage*.

Pick Most Filled metric This metric employs the intuitive strategy of selecting the most complete message for copying. In the specific case of *Sudoku* solving, this metric selects the most filled *Sudoku* partial solution. Put in other words, it selects the *Sudoku* partial solution with the lesser number of blank cells.

Pick Among First metric We know that, in some settings of collaborative problem-solving, human subjects are likely to copy the first (from left to right) solutions on the graphic interface (Farenzena *et al.* 2011). The authors have provided us with a mathematical model of this behaviour, stated as $\langle X(k) \rangle = (1 - p)^{k-1}p$, where the parameter p is fixed as $p = 0.5479$ and $\langle X(k) \rangle$ denotes the probability of an agent copying the k_{th} neighbour solution.

We inserted this behaviour into our model by firstly generating a random ordering of neighbours for each agent in order to compose a simulated graphic interface. As a result,

each agent visualizes some neighbours before or after others. Secondly, we translated the above mathematical model into a message-selecting metric in which the solution selected by an agent is the k_{th} with probability $\langle X(k) \rangle$.

Cognitive Stage

Sudoku solving techniques are abundant in *Sudoku* strategy literature. Davis (Davis 2007) discusses a collection of these techniques, of which the *Naked Singles* rule, the *Hidden Singles* rule and the *Naked Twins* rule are some examples. These techniques intend to, given a partial *Sudoku* solution, generate a set of *movements* which can be used to mark cells of this *Sudoku* puzzle. We implemented 5 of these rules, modelling each one of them as a function that maps a *Sudoku* partial solution to a set of *movements*. These functions can be used in the *Cognitive Stage* to add local changes to the *Sudoku* message received in the *Social Stage*.

The rules implemented were the *Unique Missing Candidate* rule, the *Naked Singles* rule, the *Hidden Singles* rule, the *Two out of Three* rule and the *Naked Twins* rule, all of which (with the exception of the *Two out of Three* rule) are discussed in (Davis 2007). In our modelling, each agent knows a particular quantity of rules, this quantity determining the agent's *level*. Our tests were all conducted with a heterogeneous population of agents of different *levels*. We detail below the 'two out of three' rule. For details on the other 4 rules, see (Davis 2007).

Two out of Three Rule This rule applies to groups of three contiguous 3×3 blocks. It aims to find a value v such that v is present in two of the three 3×3 blocks encompassed by the group, but missing on the third. It proceeds by enumerating all the candidate cells - namely the empty cells - on this block, and then, by eliminating from this set all the cells that are encompassed by the rows or columns in which v is placed on the other two blocks. If the resulting set has cardinality 1, the rule has successfully found a cell to mark.

Copying Solutions

In a study conducted by (Farenzena *et al.* 2011), subjects were invited to solve *Sudoku* puzzles and share them with other subjects with whom they were connected through a network topology. At any time, a player could copy solutions from one of his or her neighbours. The authors' analysis has shown that there is a different copy frequency associated with each topology. For instance, the scale free topology with $\gamma = 1.65$ resulted in a copy frequency of 87%, while the fully connected topology resulted in a copy frequency of 42%.

We incorporated this copying frequency into our modelling, recreating the scenario of the experiments conducted by (Farenzena *et al.* 2011) by setting this parameter accordingly to the topologies we used.

Conformist Behaviour

We introduce in our modelling the *conformist* behaviour proposed by (Efferson *et al.* 2008) and further studied by (Farenzena *et al.* 2011), which states that agents are likely

8	1	2			4			3
3	7	9	8		2	1	4	
5	6	4	3	9	1	2	8	7
9	4	1		8	5	3	7	
7			9	4	3	8		1
	8	3	1		7	4	5	9
		7	4	3	9		1	8
1	3	8			6			4
4	9			1	8		3	

Figure 1: A *Sudoku* puzzle solvable only by trial-and-error

to copy a particular solution when surrounded by neighbours that share this solution.

Our approach was to establish a *threshold* determining a minimum necessary quantity N of neighbours sharing a particular solution. In our modelling, all agents with N or more neighbours having a solution s in common are programmed to copy solution s from them.

Guessing and Backtracking

We have consistent evidence that trial-and-error is a part of the *Sudoku* solving experience. The need for trial-and-error in *Sudoku* puzzles is not a falsifiable conclusion, but a mathematical fact (Davis 2007). Some puzzles are only solvable by the means of a backtracking procedure (see Figure 1).

In our modelling, we associate each agent with a numerical parameter determining the probability this agent has to guess when incapable of applying a typical *Sudoku* solving strategy.

Automatic *Sudoku* solvers employing a backtracking algorithm are easily programmed and very time-efficient. On the other hand, the space complexity of these algorithms is a barrier to most human solvers, who need to write down tons of observations in order to employ a backtracking strategy. With this in mind, we propose in our modelling two different kinds of backtracking, intended to be more similar to the way human beings employ error correction in *Sudoku* solving in a collaborative environment such as the one analysed in (Farenzena *et al.* 2011).

- **Local Backtracking:** When faced with one or more conflicts in its own solution, an agent simply erases the conflicting cells.
- **Social Backtracking:** When faced with one or more conflicts in its own solution, an agent copies a solution from one of its neighbours. In our modelling, this is done by raising the copy rate of this particular agent to 1.

Results and Analysis

We tested the above modelling using an heterogeneous, uniform agent distribution in which every agent *level* (0 to 5) had an equal representation in the population. We performed

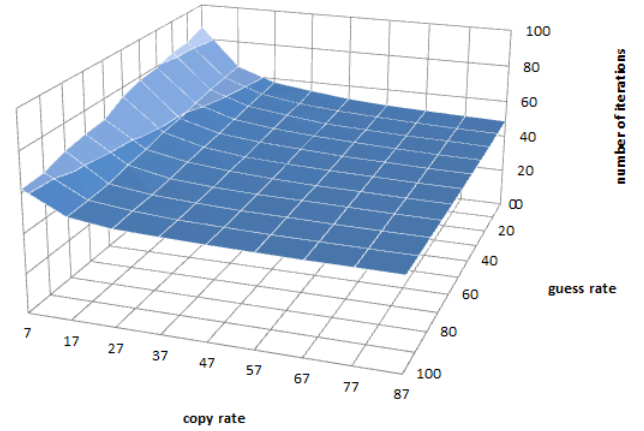


Figure 2: Effects of copy rate and guess rate on the algorithm's performance (scale-free topology with 'pick most filled' selection metric)

tests with varying values for the parameters of *copy rate*, *guess rate*, *message selecting metric* and *network topology*.

Copying and Guessing are Beneficial to the Algorithm's Performance

Farenzena (Farenzena *et al.* 2011) have stated that, in their settings, "*cooperation works because agents can copy each other*". This was our hypothesis, and our experiments indicate that, indeed, in some of the topologies tested, the *copy rate* has a decisive role in enhancing the algorithm's performance (see Figure 2 and Figure 3). The graphs in Figure 2 and Figure 3 - in which the vertical axis represents the number of iterations needed for the agents to solve the problem - both show a maximum in (*copy rate*: 7, *guess rate*: 0) and a minimum in (*copy rate*: 87, *guess rate*: 100), indicating that the greater the *copy* and *guess* rates are, the fewer iterations are needed for the agents to solve the problem. We understand this is due to the fact that some agents of low levels are incapable of solving some instances of the *Sudoku* problem, don't knowing the *Sudoku* solving rules needed to mark its cells. These agents have no hope solving the problem without some guessing mechanism, and take advantage of high *copy rates*, which can be used to correct wrong guesses through copying better agent's solutions. Also, as shown in Figure 4, there is a correlation between the copy and guess rates and the convergence of the algorithm: the greater they are, the greater is the probability that the agent network will be able to solve the problem.

Evaluating the Neighbours' Solutions is Beneficial to the Algorithm's Performance

Of the three metrics employed by us in the *Social Stage*, only one had the agent evaluate the content of the solutions proposed by their neighbours. Another chose who to copy based on position, simulating observed human behaviour (Farenzena *et al.* 2011), and the remaining one chose a neighbour randomly. Our hypothesis was that the first metric would

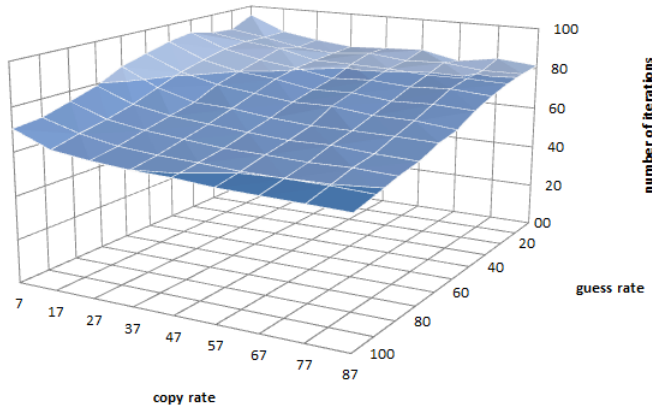


Figure 3: Effects of copy rate and guess rate on the algorithm's performance (scale-free topology with 'pick among first' selection metric)

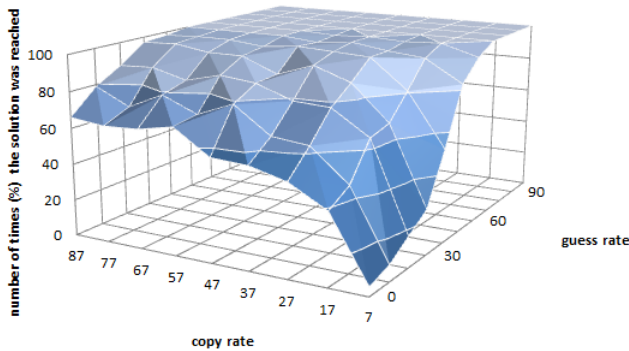


Figure 4: Effects of copy rate and guess rate on the algorithm's convergence (scale-free topology with 'pick among first' selection metric)

perform better than the other two, a result which would confirm our belief that problem-solving social networks are able to respond positively to the employment of evaluation functions.

As can be seen in Figure 5, our experiments have shown that evaluating the proposed solutions indeed bring significantly better results, despite the fact that humans seemingly avoid that strategy (Farenzena *et al.* 2011). In all topologies, choosing the solution with the largest number of filled cells not only allows agents to solve the instance earlier, but also causes the network to converge faster to the correct solution. That is, once one individual has found the correct solution, that solution spreads faster through the network if the other agents are evaluating the solutions that reach them.

The strategy based on the behaviour observed in humans failed roughly the same as choosing a random neighbour to copy, and both underperformed in comparison to evaluating the solutions according to number of filled cells. This confirms that choosing solutions arbitrarily without taking their content into consideration is prejudicial to the process of problem-solving.

Even so, that behaviour was observed in human networks. Farenzena (Farenzena *et al.* 2011) has suggested that human beings behave like that because in the real world the environment continuously selects against ineffective solutions, since a bad choice in solving a real world problem might cause and agent to cease communication. Since neither the original experiments nor our simulation included such a mechanism, the advantages of that process have not been observed, but they might already be good enough that it may not justify the effort for the agents to evaluate the solutions.

Another explanation might be that humans might find difficult evaluating sudoku partial solutions. Although we have used the number of filled cells as a metric, that is not guaranteed to accurately measure which solution is best, since individuals might fill in cells with the wrong value. That hypothesis is supported by the experiments of (Mason and Watts 2011), which used optimization of multidimensional functions as the problem being solved. This problem has a straightforward and accurate method of determining the fitness of a solution, which is simply the value of the given function for the proposed coordinates. In this experiments, the subjects seemingly did effectively evaluate the proposed solutions, giving preference to the ones that were better evaluated. The difference in behaviour between the two experiments might be because in the latter the fitness of a certain solution was obvious to the agent, while in the former it's not as easy to pick the best solution out of the available ones.

Small Changes on the Interface Might Improve the Performance of Human Beings in Human Computation Environments

Knowing that, systems that take advantage of human computation might take advantage of the computer aspect of human computation to encourage selection of better-evaluated solutions. One possibility is ordering the neighbours according to the quality of their solutions, with better-evaluated ones being displayed first, instead using a pre-specified or random order. However, that might introduce a (BLANK) in

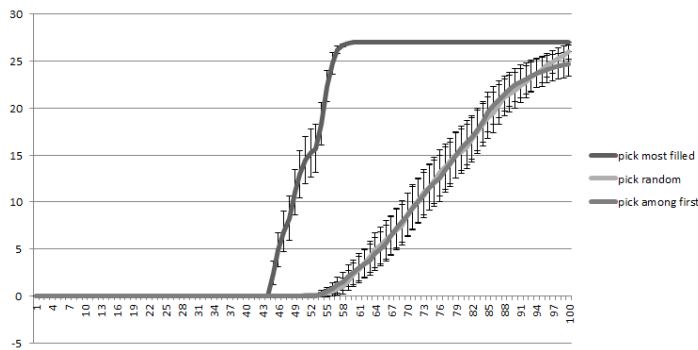


Figure 5: Progression of the solution. The graph shows how many agents have successfully solved the puzzle in a given iteration (scale-free topology)

the system, while also limiting variability and discouraging individuals that might have eventual more effective evaluation techniques. To avoid that, perhaps simply displaying the evaluation of each solution might be enough to encourage that behaviour.

Conclusions

We compared results from social computing experiments done with human subjects with agent-based simulations mimicking the same problem-solving environments. To do this, we proposed a novel method for artificial social problem-solving and used it to model the *Sudoku* problem-solving environment studied by (Farenzena *et al.* 2011). We reached important conclusions concerning the dynamics of problem-solving collaborative environments: the behaviours of copying and guessing are shown to be beneficial to these systems. We also discuss the benefits of the behaviour of evaluating neighbours' solutions in order to select one of them for copying, rather than simply selecting a random solution or even the most readily available one, as the analysis conducted by (Farenzena *et al.* 2011) has shown to be the case sometimes. Thus, having in mind the fact that the strategies employed by human subjects are often not the most efficient, we proposed a collection of guidelines concerning the design of human computation systems, which intend to improve the performance of the human subjects they employ.

References

- Luis Von Ahn and Laura Dabbish. Designing games with a purpose. In *Communications of the ACM*, volume 51, pages 58–67, August 2008.
- Ricardo Araujo and Luis Lamb. Memetic networks: analyzing the effects of network properties in multi-agent performance. In *AAAI'08*, pages 3–8, 2008.
- Seth Cooper. Algorithm discovery by protein folding game players. In *National Academy of Sciences*, volume 108, pages 1–5, June 2011.
- Tom Davis. The mathematics of sudoku. November 2007.
- Richard Dawkins. *The Selfish Gene*, volume 32. Oxford University Press, 1976.

C. Efferson, R. Lalive, P.J. Richerson, R. McElreath, and M. Lubell. Conformists and mavericks: the empirics of frequency-dependent cultural transmission. In *Evolution and Human Behaviour*, volume 29, pages 56–64, 2008.

Daniel Farenzena, Ricardo Araujo, and Luis Lamb. Collaboration emergence in social networks with informational natural selection. In *2011 IEEE Third Intl Conference on Privacy Security Risk and Trust and 2011 IEEE Third Intl Conference on Social Computing*, pages 555–559, 2011.

Michael Kearns. Experiments in social computation. In *Communications of the ACM*, volume 55, pages 56–67, June 2012.

Winter Mason and Duncan J. Watts. Collaborative learning in networks. In *Proceedings of the National Academy of Sciences*, volume 2011, pages 764–769, November 2011.