Proceedings, The Third AAAI Conference on Human Computation and Crowdsourcing (HCOMP-15)

From "In" to "Over": Behavioral Experiments on Whole-Network Computation

Lili Dworkin and **Michael Kearns**
University of Pennsylvania
ldworkin@seas.upenn.edu, mkearns@cis.upenn.edu

Abstract

We report on a series of behavioral experiments in human computation on three different tasks over networks: graph coloring, community detection (or graph clustering), and competitive contagion. While these tasks share similar action spaces and interfaces, they capture a diversity of computational challenges: graph coloring is a search problem, clustering is an optimization problem, and competitive contagion is a game-theoretic problem. In contrast with most of the prior literature on human-subject experiments in networks, in which collectives of subjects are embedded "in" the network, and have only local information and interactions, here individual subjects have a global (or "over") view and must solve "whole network" problems alone. Our primary findings are that subject performance is impressive across all three problem types; that subjects find diverse and novel strategies for solving each task; and that collective performance can often be strongly correlated with known algorithms.

Introduction

There is a long line of behavioral experiments that examine computation and strategic interaction in networks (see e.g. Kearns, Suri, and Montfort 2006; Kearns 2012 and the references therein; Suri and Watts 2011; Mao et al. 2011). These studies investigate collective problem-solving in networks when subjects are given control over a single vertex and only allowed to observe and interact with immediate neighbors. The experiments proceeded simultaneously, with subjects working in parallel (often colocated in a computer laboratory) to solve some collective task. Overall, these studies demonstrate that human subjects perform well on such decentralized network problems, and are able to solve challenging coordination and computational tasks.

Our setting is related but different. We provide each subject with a *global* view of the network, and allow control of all vertices. Each individual must complete a series of whole-network tasks on a web-based application. Our motivation for shifting from local to global information and tasks stems from the increased societal awareness of networks in everyday life: individuals no longer simply act *within* networks, but have begun to reason *about* them as well (Giles 2010). A number of studies suggest that social networks have significant effects on multiple aspects of well-being, including health and happiness (Christakis and Fowler 2007; Fowler et al. 2009; Centola 2010). Furthermore, there are now a variety of online tools available that allow visualization and interaction with one's social networks.[1] Given the increasing ubiquity and awareness of this abstraction, there is scientific value in discovering how people understand, reason about, and perform computations on networks.

Although still a relatively new area of study, there is interesting prior work on the extent and effects of such "network cognition." A study by Dessi et. al. (2012) tested the ability of individuals to memorize information about the degree distributions of social networks. Recent experiments by Bannerjee et. al. (2014) asked subjects to identify the individuals who are most "central" in their social networks. Kilduff and Krackhardt have investigated the effects of network perception within organizations (2008).

We study three rather different network tasks: graph coloring, graph clustering, and competitive contagion. The goals, respectively, are: find a proper coloring as quickly as possible; cluster vertices (using as many or as few clusters as desired) to maximize a standard objective function (the Q-score of Newman and Girvan 2004); and choose two seed vertices to maximize the spread of infection (or "market share") at the expense of your opponents. This suite of tasks has the nice property that while each can be presented to subjects with similar interfaces and action spaces (the selection of colors for vertices), they also capture a diversity of computational challenges: graph coloring is a search problem, the graph clustering task an optimization problem, and competitive contagion is a game-theoretic problem.

We note that many previous studies of human computation and crowdsourcing focus on tasks that are relatively easy for humans, such as image labeling or document sentiment analysis (see Yuen, King, and Leung 2011 for a survey of commonly studied tasks). In contrast, here we focus on tasks known to be computationally challenging — all three are formally intractable (NP-hard) for large worst-case instances. We thus have no reason a priori to believe that humans will be able perform well. While there is prece-

Copyright © 2015, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

[1] See, for instance, the TouchGraph Facebook Browser (http://www.touchgraph.com/facebook) or Mentionmapp for Twitter (http://mentionmapp.com).

	Graph Coloring	Graph Clustering	Competitive Contagion
Task Type	search	optimization	game-theoretic
Task Objective	minimize time to properly color	maximize Q-score	best response to population distribution
Graphs Used	generative models	generative models, real-world social networks	generative models, theoretically-inspired networks
Algorithmic Benchmarks	backtracking, simulated annealing	Newman heuristics	degree-greedy

Table 1: Summary of the three experiments.

dent for using human workers to perform challenging tasks, such as drawing (Koblin 2009) or peer review in online classes (Suen 2014), our emphasis is rather different. We focus on computational problems with clear input and output, rather than creative or subjective tasks. An example of more relevant prior work is Foldit, an online multiplayer game in which participants solved complex algorithmic protein structure problems (Cooper et al. 2010).

In each of our experiments, a set of about 30 graphs were presented to a population of about 100 subjects, each of whom was required to solve the task on every graph in the set. In general, performance was impressive. Subjects found proper colorings for complex graphs in only three minutes on average, and the best players required less than one hundred seconds. A large fraction of subjects were able to find optimal clusterings on *every* graph presented to them. The population as a whole came close to achieving equilibrium on most graphs in the competitive contagion task.

In addition to reporting on the strong collective performance, we study relations and correlations with known heuristics, the diversity of individual strategies, and the effects of network structure. Key findings include:

- **Coloring:** A small number of players were notably better at the given task than the average. Graphs with high-degree vertices and many non-isomorphic solutions were easier to solve. Players exhibited a diversity of greedy and other strategies.
- **Clustering:** Many subjects consistently outperformed known heuristic algorithms. The use of the web application's layout rearrangement feature was crucial, and players took advantage of this functionality in different ways.
- **Contagion:** The population came close to equilibrium on graphs with dominant strategies, but had more difficulty on graphs that require the coordination of diverse strategies, such as those with multiple components.

In the remainder of the paper, we first describe the common experimental design, and then consider each experiment in turn. Taken together, the results suggest inspiring subject aptitude for individual problem-solving on networks.

Experimental Design and Methodology

Our primary subject population was 104 students taking an undergraduate course on the science of social networks at the University of Pennsylvania. Being a survey course, the population included a healthy mixture of majors (both quantitative and non-quantitative) and significant representation of all four graduating classes.[2] Each student was required, as a class assignment for credit, to solve the task on every graph in a collection. Students were clearly told to work alone, and that their grade on the assignment would be determined exclusively by their actual performance on the task; they thus had strong incentives to perform each task well. In our analyses, we discard any subjects who failed to submit a solution for every graph, and therefore the effective population sizes range from 91 to 104. Some of the tasks were also performed by a population of AAAI conference attendees, as well as a class of students taking a MOOC version of the class on Coursera. For our numerical analyses, we use data from our academic institution subjects only (though we emphasize that numerical results are very consistent across all populations). In some cases, we draw on the secondary populations for anecdotal subject-level findings and strategies.

Each experiment was conducted over the course of about 1-2 weeks. Subjects were given access to a web application to which they could log on at their convenience (see Figure 1 for screenshots). The landing page described the task and incentives or objective, but provided no guidance on how to approach the task. Upon log in, subjects chose a graph to work on from a drop-down menu. Each task could be accomplished by using the application to change the color of vertices: e.g. in the clustering task, choosing a vertex's color corresponded to assigning its cluster, and in the contagion task, choosing a vertex's color corresponded to selecting it as a seed. The application also allowed subjects to drag-and-drop vertices to rearrange the layout.

Although all graphs had to be completed by the end of the allowed period, the subjects were not required to complete all graphs in a single session. For clustering and competitive contagion, subjects were also allowed to resubmit solutions as often as desired in an attempt to improve their score, which was based on their last submission. At the end of each task, subjects filled out a survey asking them to discuss the strategies they used to solve the problem.

Graph Coloring

In this task, subjects were required to find proper graph colorings; that is, to assign colors to the vertices of a graph so

[2] In particular, the class consisted of 19% freshmen, 11% sophomores, 28% juniors, and 42% seniors. The three most popular majors were Systems Science and Engineering (21%), Computer Science (13%), and Networked and Social Systems Engineering (12%). The class was 70% male and 30% female.

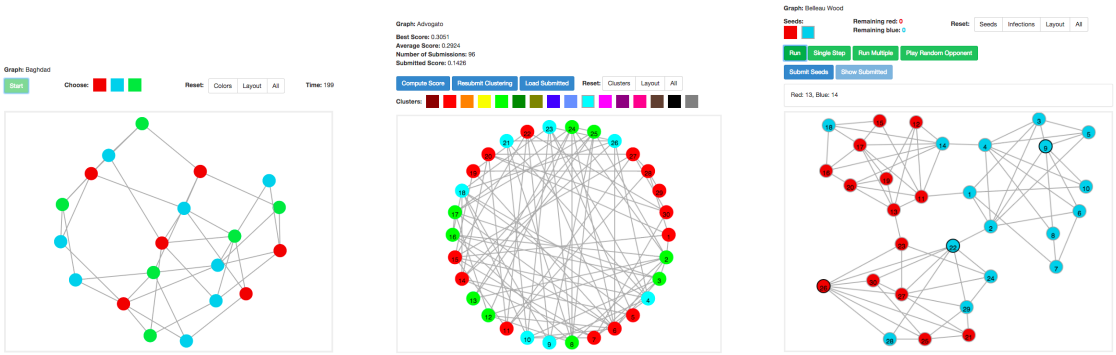


Figure 1: Sample screenshots of the web applications for the coloring, clustering, and competitive contagion tasks, respectively. While some auxiliary application features varied between the experiments, the main functionality — which allowed vertices to be rearranged and colored — remained the same.

that no two adjacent vertices share the same color. The subjects were given a number of colors equal to the chromatic number of the graph, and therefore were not required to perform any optimization, only search. The goal was to find a solution as fast as possible, and the assignment was graded to incentivize this. While the subjects were given unlimited attempts to solve the graph, once a successful coloring was submitted, no further attempts were considered. This was done to prevent students from memorizing a solution and then improving their speed in a subsequent trial. In the following sections, we use the term "solution time" to refer to the elapsed time of a subject's first successful coloring attempt on a graph.

We generated ten of the 30 graphs used in the experiment from an Erdős-Rényi model (Gilbert 1959); another ten from the Watts and Strogatz small-world model (Watts and Strogatz 1998); and the last ten from the Barabási-Albert preferential attachment model (Barabási and Albert 1999). The parameters of each generative model were chosen (and additional modifications made) so that each graph contained 20 nodes and 40 edges.

Collective Performance

In general, subjects found proper colorings quickly. The average solution time across all subjects and all graphs was 182.75 seconds, and all but two subjects had average solution times below five minutes. There were also a small number of players who performed significantly better than the average. The three fastest players had average solution times of 74.65, 77.75, and 95.63 seconds. Furthermore, these three players actually solved *each* of the 30 graphs faster than the average solution time on that graph. The coloring task is in fact the only task in which we can clearly distinguish consistently superior players. We note that a key difference between this task and the others is that subjects had no opportunity to improve their performance. Once a player solved a graph, any subsequent faster attempts were ignored. Thus, while the clustering and contagion tasks rewarded practice and effort, the coloring task rewarded some innate aptitude.

Effects of Network Structure

While performance was impressive overall, there was disparity of difficulty across the different types of graphs. The small-world graphs were the hardest for subjects to solve, with an average solution time of 208.27 seconds; Erdős-Rényi graphs ranked next, with an average solution time of 172.89 seconds; and preferential attachment graphs were the easiest, at 163.55 seconds. (All 6 pairwise comparisons between graph families pass t-tests for differences of means with $p < 0.03$.) Additionally, the correlation between the average degree of the five highest degree vertices in a graph and the average solution time for that graph is -0.37 ($p \approx 0.04$), suggesting that graphs with vertices of high degree are easier to solve. This observation is consistent with the graph family difficulty rankings, as preferential attachment graphs contain the vertices of highest degree, followed by Erdős-Rényi, and then small-world. One hypothesis is that graphs with high-degree vertices suggest a natural coloring algorithm; namely, begin by coloring all "hub" nodes, and then fill in the gaps. Indeed, as we see shortly, many subjects did use some variant of this strategy.

We also observe that graphs with more possible solutions (i.e. non-isomorphic proper colorings) were easier for the subjects. Because it is computationally intractable to compute this number even on small instances, we use as a proxy the number of different solutions found by the population. The correlation between the number of solutions found for a graph and average solution time for that graph is -0.49 ($p \approx 0.006$). Indeed, the population was able to find only one solution for the graph with the highest average solution time (497.19 seconds compared to the average over all graphs of 182.75).

Comparisons to Algorithmic Benchmarks

It is interesting to investigate whether subjects collectively behave in ways that reflect standard heuristic algorithms for a given task. We implemented a backtracking algorithm that colors nodes greedily in decreasing order of degree, and counted the number of steps required to find a solution on each graph. We also implemented a simulated annealing algorithm (Kirkpatrick, Gelatt, and Vecchi 1983) for graph

coloring, where the state space is the set of possible colorings and the cost function is the number of edges colored incorrectly, and again counted the number of steps required to find a solution (averaged over 100 trials, since simulated annealing is a randomized algorithm). The correlations between these counts and average population solution time are 0.58 and 0.65 respectively ($p < 0.001$). These results indicate that graphs that are easier for heuristics to solve are also easier for the players, perhaps due to the structural properties (discussed above) that affect the difficulty of the task for *both* algorithms and humans. Of course, these correlations do not suggest that the human subjects are necessarily using one of these two heuristics to color the graphs. Yet as we see later, there is reason to believe that subjects adopted strategies with a backtracking "style", in the sense that they colored greedily and preferred high-degree vertices.

Individual Strategies

We find evidence that many subjects converged on one of two broad strategies for coloring a graph: "color-greedy", i.e. choose a color swatch and then color as many nodes as possible with that color before switching; or "neighbor-greedy", i.e. choose a swatch, color a node, choose a new swatch, color a neighboring node, etc. To measure the extent to which a given player aligned with one of these strategies, we computed the following two quantities (averaged over all graphs). First, we counted the number of times the subject chose a new color swatch, and then divided by the total number of coloring steps, to obtain a "color-changes-per-step" value. Second, we counted the number of times a subject colored two neighboring vertices one immediately after the other, and then divided by the total number of coloring steps, to obtain a "neighbor-colorings-per-step" value. Subjects adopting a color-greedy strategy will tend to have relatively low color-changes-per-step; subjects adopting a neighbor-greedy strategy will have relatively high neighbor-colorings-per-step.

Table 2 illustrates these two styles of play. We identify the players with the highest and lowest color-changes-per-step values and compare screenshots of their play on a particular graph. Both of these players are among the top ten performers in the population, but consistently use distinctly different strategies. To quantify their styles by the metrics above, we note that the color-greedy player has a color-changes-per step value of 0.31, whereas the neighbor-greedy player's value is 0.82. The color-greedy player's neighbor-colorings-per-step value is the lowest of all players, at 0.17. The neighbor-greedy player's value is the third highest, at 0.56. While these two players are canonical examples of each style of play, we emphasize that neither is alone in his adoption of a particular strategy. Indeed, many subjects consistently used one or the other on all graphs.

Regardless of preferred strategy, most players gave preference to high-degree vertices when choosing which vertices to color first. For each player and each graph, we divided the degree of the first vertex colored by the highest degree in the graph. The average of this value over all players and graphs is 0.73. Furthermore, on 37% of all coloring attempts, the player colored the highest degree vertex first.

We further note that there are two variants of the neighbor-greedy strategy. Many players primarily identified triangles in the graph, and colored each of the three nodes using a different swatch. Other players primarily identified nodes of high degree, and colored these hub nodes first, followed by their neighbors. There is anecdotal evidence from the survey responses that many players began by using a degree-greedy approach, and then switched to a triangle-greedy approach. One subject commented: *"First I was starting with the vertex of the highest degree and alternating the colors of its neighbors. I realized that wasn't working at a certain point and then started using the triangle strategy." Another remarked: "I tried coloring the vertices with the largest degrees first and then coloring the rest of the vertices from the inside out. This was also not ideal. ... Lastly, I tried the triangle method ... While it wasn't a fool-proof strategy, it worked on the first try many times and was more effective."*

We identified at least one player who successfully adopted a coloring strategy radically different from those above. Whereas most players made little use of the drag-and-drop feature of the application, and rearranged the layout only to remove occlusions, this player's strategy relied entirely on manipulation of the layout. The subject would first rearrange the vertices into columns in which there were no edges within each column — in other words, a k-partite graph, where k is the chromatic number. After rearranging the layout, she generated the coloring in batches columnwise. See Table 3 for an illustration. We find it interesting that the strategies used by the subjects generally fall into identifiable classes, suggesting that there are a few distinct ways that humans solve each problem.

Graph Clustering

The next task required subjects to solve the problem of community detection in networks. The objective was to organize vertices into "clusters," with many edges connecting vertices within each cluster, and few edges between vertices in different clusters. Subjects were responsible for both deciding the number of clusters to use, and determining the assignment from vertices to clusters. While there are many ways to measure the quality of a clustering, we score the subjects according to the well-known *modularity* metric proposed by Newman and Girvan (2004). The modularity of a graph is defined by $Q = \sum_{i=1}^{k}(e_{ii} - a_i^2)$, where i ranges over all k clusters, e_{ii} is the fraction of edges with both end vertices in the same cluster i, and a_i is the fraction of edges that are attached to cluster i. Intuitively, this metric favors a clustering in which inter-cluster edge densities are high compared to between-cluster edge densities. The best possible Q-score varies depending on the graph, as those with more "community structure" will have higher possible values. A value of 0 indicates that the clustering is no better than random, and can be trivially achieved by assigning all vertices to one cluster. Typical values for real-world networks tend to fall between 0.3 and 0.7 (Newman and Girvan 2004).

The web application allowed subjects to try out different clusterings and see the corresponding Q-scores. Subjects were allowed to resubmit clusterings as often as desired over

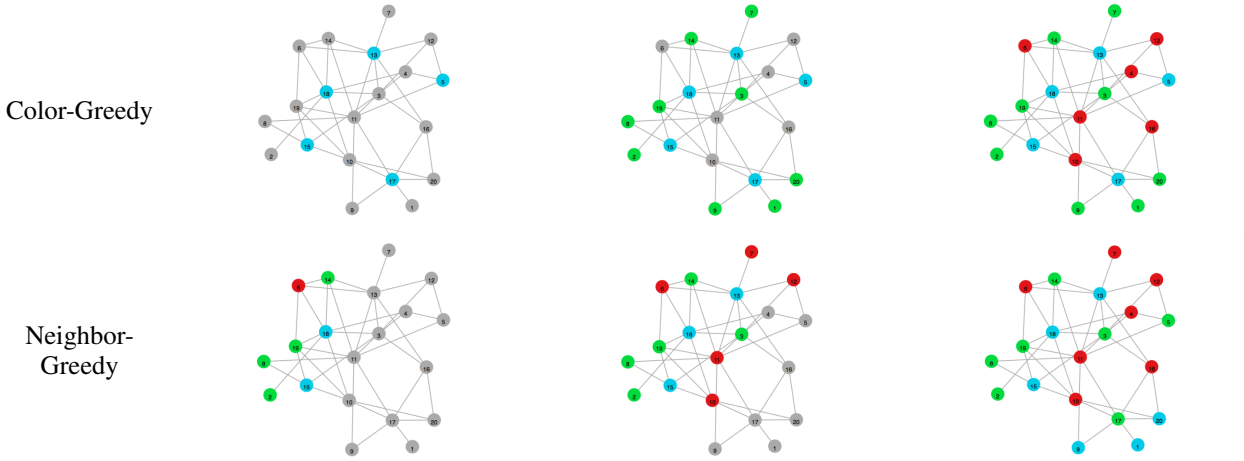


Table 2: Illustration of the color-greedy and neighbor-greedy coloring strategies. We identify the two players with the highest and lowest color-changes-per-step values (0.82 and 0.31, respectively) and compare temporal snapshots on their play on a particular graph. The color-greedy player first colored as many vertices as possible with the blue swatch, then switched to green, and completed the coloring with red. The neighbor-greedy player began coloring on the left side of the graph, then moved gradually and locally to the upper right, and finished the coloring on the bottom right.

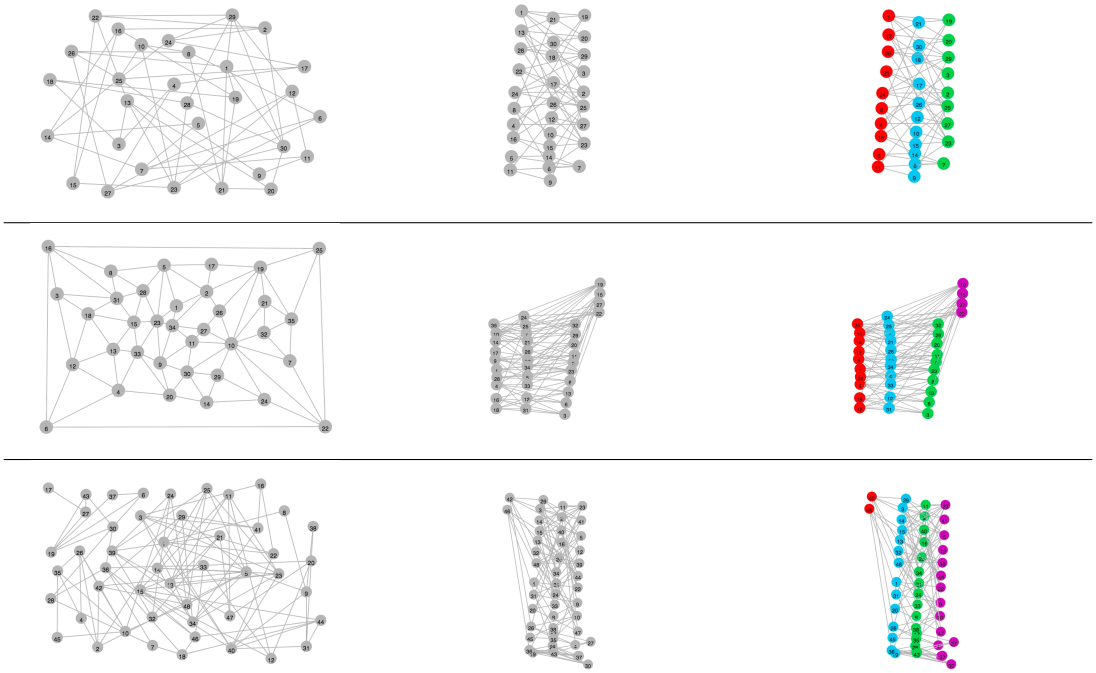


Table 3: Illustration of a column playing style. This player first rearranged the graph into columns corresponding to independent sets, and then colored in batch. We show temporal snapshots of the subject's play on three different graphs.

the course of the experiment. The application also displayed the highest Q-score achieved by any subject so far on each graph. Thus, unlike graph coloring, the objective function here was the quality of the solution found (optimization), rather than the time to find a solution (search).

There were 28 graphs in total, of which 23 were generated so as to have a "ground truth" clustering. Each of these graphs was composed of 2-4 Erdős-Rényi components, where the within-component edge densities were chosen in $[0.5, 0.7]$ and between-component edge densities were chosen in $[0.05, 0.07]$. Each graph contained 30 vertices. The remaining six graphs were subgraphs of "real-world" networks from the KONECT and UCINET datasets (Kunegis 2013; Borgatti, Everett, and Freeman 1999).

We note that, for the other two tasks, we used a standard force-directed layout algorithm to display the graphs. But because this algorithm often makes clustering structure visually obvious, for this task we chose instead to display every graph in a random circular layout. Thus, the drag-and-drop functionality of the application proved to be a crucial asset, and most players took advantage of the feature to rearrange vertices into clusters. We provide further discussion of the use of this feature in a subsequent section.

Collective Performance

To quantify performance, we might compare the Q-scores achieved by the subjects to the maximum possible Q-score on each graph. However, it is an NP-hard problem to find the Q-maximizing clustering, and intractable even on small instances. For measuring relative performance, however, we can use the best score achieved by any member of the population on each graph as a proxy. More formally, if Q_{ij} is the Q-score achieved by player i on graph j, we let $Q^*_j = \max_i Q_{ij}$. One might expect that a better approximation could be achieved by a known heuristic, but as we see in the next section, this is not the case; the best players consistently outperformed two different heuristics.

Let $S_{ij} = Q_{ij}/Q^*_j$ denote the relative performance of player i on graph j, and let $S_i = \text{avg}_j(S_{ij})$ denote player i's average relative performance over all graphs. There were 22 players who achieved the maximum Q-score on *every* graph and therefore have $S_i = 1$. The average S_i value over all players is 0.96, and over 80% of players have values above 0.95. These results suggest that not only do the best players perform consistently well, but in fact almost all players found nearly-best solutions on the majority of graphs.

Comparison to Algorithmic Benchmarks

We compare subject performance to that of two algorithms for finding approximately optimal Q-maximizing partitions, which we denote "Newman-fast" (Newman 2004) and "Newman-eigenvector" (Newman 2006). On 10 of the graphs, the algorithms found the same clustering and therefore achieved the same Q-score. On 12 of the graphs, Newman-fast found a better clustering than Newman-eigenvector, and on the remaining six, the opposite occurred. As might be expected, the scores of the two algorithms across graphs are highly correlated (0.93, $p < 10^{-12}$).

On 11 of the graphs, the maximum Q-score achieved by any player (Q^*_j) *exceeded* the maximum score found by the heuristics. On all remaining graphs, the maximum player score matched the maximum heuristic score. As noted previously, there are 22 players who achieved the maximum Q-score on every graph, and therefore each of these subjects consistently outperformed the heuristics. We note that on 10 of the 11 graphs on which the best player's score exceeds the best heuristic's score, the two heuristics also disagree on the optimal clustering. Thus, it seems there is something intrinsically difficult for the heuristics about these graphs, but which does not pose a problem for the best human subjects.

To determine whether population and heuristics agreed on the relative difficulty of the graphs, we first calculated average population performance on each graph, namely $G_j = \text{avg}_j(S_{ij})$. We then divided each heuristic's Q-score by the approximately optimal score Q^* to determine the heuristic's performance on each graph. When we correlate these vectors, we obtain coefficients of 0.62 and 0.61 for Newman-fast and Newman-eigenvector, respectively ($p < 0.001$). Thus, the population as a whole performed better on graphs for which the heuristics were able to find better clusterings. This finding echoes those for coloring with backtracking and simulated annealing.

Individual Strategies

As in the coloring experiments, we can identify diverse individual styles of play. In particular, we observed the most striking variability in the use of the application's drag-and-drop functionality. As previously mentioned, many of the players rearranged vertices (which were by default displayed in a circle) in order to expose the clusters. Yet while some players made all of their display changes before coloring vertices (i.e. assigning clusters), others interleaved display changes and color changes, and others rearranged the vertices only after settling on a clustering. To quantify this for a particular player on a particular graph, we compute the mean index of the player's coloring moves, subtract the mean index of display change moves, and divide by total number of moves. So for example, if a player first rearranged all vertices, and then clustered all vertices, this value is 0.75 - 0.25 = 0.5. If a player alternated between display changes and cluster changes, the value is 0.5 - 0.5 = 0. There is a great deal of variety in the average "style" value for each player, which ranges from -0.44 to 0.44. Players with strongly positive or negative values use a distinct two-phase strategy, in which they either first rearrange the vertices into a visual partitioning and then assign clusters, or the reverse. There are also some players who often made *no* changes to the display, and for whom this value is undefined.

Table 4 illustrates these three styles of play, which we denote "layout-first", "color-first," and "color-only." We identify the two players with the highest and lowest style values, and also one for whom the value is undefined, and compare screenshots of their play. All of these players are high performers, with $S_i > 0.98$, but they take notably different approaches to the problem. For confirmation from the player themselves, we turn to the survey responses. The layout-first player commented: *"It was impossible to tell what was*

clustered without moving items around. I bunched nodes by clusters." The color-first player disagreed about this impossibility, and remarked: *"Once I had identified clusters I usually moved vertices to their respective cluster groups to make it easier to visualize."* The color-only player said *"I just tried to use the list of adjacent vertices to choose how to group a set of vertices"*, and answered that she only used the drag-and-drop feature occasionally.

The majority of players preferred to use the layout-first style. While only four players have a style value of less than -0.2, 39 players have a value greater than 0.2. Although neither strategy consistently outperformed the other, we do find that use of the drag-and-drop feature (regardless of when) is correlated with a player's performance. In particular, the correlation between the number of vertex rearrangements that a player made and the player's S_i performance value is 0.24 ($p \approx 0.02$).

Competitive Contagion

Our last task is based on the model of networked competitive contagion developed in (Goyal and Kearns 2012), which built upon the work of (Bharathi, Kempe, and Salek 2007; Borodin, Filmus, and Oren 2010; Chasparis and Shamma 2010). In this model, there are two competing players, denoted "Red" and "Blue." Each player is allowed to choose two "seed" vertices, which are then infected with his color. (If a vertex is chosen by both players, its color is chosen at random.) After the initial seeding, stochastic adoption dynamics determine the spread of each infection. We use a discrete time model. On each step, we consider all uninfected vertices that are adjacent to an infected vertex. If such a vertex has more Red neighbors than Blue neighbors, it becomes Red, and similarly for Blue. If a vertex's neighbors are evenly split, the color of the vertex is chosen at random. At the end of the process, every vertex in the connected component of a seed will be infected. The goal of each player is to choose seeds to maximize eventual adoption throughout the graph at the expense of its competitor.

Rather than having pairs of players compete directly, we use a design in which each subject competes against the current *population distribution.* Each subject was required to choose the Red seeds for a set of 42 networks. The application provided the subjects with a simulator that allowed experimentation with different choices of Red and Blue seeds, see randomized outcomes, and compute average adoptions over many trials. Importantly, the application also allowed a subject to play against the seed choices already submitted by other members of the population (in the Blue role). Thus by pressing a "Play Random Opponent" button, a player could (repeatedly) sample a Blue seed pair from the current distribution of submissions, and then run simulations to determine which Red seeds compete favorably. This permitted a player to optimize his strategy against the current population of opponents. The subjects were also allowed to update their choices for the Red seeds as often as desired. Thus, the population evolved over time, as subjects returned to change their choices in response to the updates made by others.

We designed our scoring rules to emphasize that, from an individual player's perspective, this is a game in which the opponent is the *distribution* over all other players' choices. Let $G = (V, E)$ be a graph on which the game is played, and let $I_G(x, y)$ denote the expected fraction of Red infections when $x = (x_1, x_2), y = (y_1, y_2)$ are the initial seed pairs chosen by the Red and Blue player, respectively.[3] Let P_G denote the population distribution over seed pairs, and let U_G be the support of P_G. The *score* of seed pair x on G is then defined as $S_G(x) = \sum_{y \in U_G} P_G(y) I_G(x, y)$. In other words, the score of x is the expected fraction of infections that x wins against the population distribution P_G.

We generated 36 of the 42 graphs by composing 2-3 components, each of which comes from the Erdős-Rényi or preferential attachment model. The components are then connected by some variable number of edges. The remaining 6 graphs are instances of constructions in (Goyal and Kearns 2012) designed to elicit interesting strategic tensions. The number of vertices in each graph ranges from 16 to 60.

Collective Performance

Given the incentives described above, the right measure of collective performance is the population's distance from *equilibrium*, the state in which no player can unilaterally improve his score by changing his seed choice.[4] To quantify this distance, we introduce the concept of *regret*, which measures how close each player's score is to the best player's score. More formally, the regret R_G of the population P_G on graph G is $\sum_{x \in U_G} P_G(x)(\max_{y \in U_G} S_G(y) - S_G(x))$. It is easily verified that equilibrium is reached if and only if 1) $R_G = 0$, and 2) there exists no seed pair z such that $S_G(z) > \max_{y \in U_G} S_G(y)$. Checking the satisfaction of condition 2 is intractable in general, so we use R_G as a one-sided measure of how far the population is from equilibrium.

The average regret across all 42 graphs is 0.0129, less than 3% of the average player score of 0.4911, indicating strong collective performance by this measure. Yet if we analyze the graphs individually, we find significant variation. Behaviorally speaking, there seem to be two types of graphs. The first type of graph has a *dominant strategy*, i.e. a seed pair that always infects at least as many vertices as the seed pair it plays against. In this case, the distribution over opponents' choices is irrelevant, and the game becomes a single-player optimization problem in which the goal is to identify the dominant choice. Note that equilibrium can be achieved on this type of graph only when everyone plays the dominant strategy. Although it is again intractable to decide whether a graph has such a strategy, we can check whether the most popular seed choice is a dominant strategy. By this calculation, at least 17 of the 42 graphs do have a dominant strategy. Figure 2 illustrates such a graph.

Graphs of the second type have no dominant strategy, and so the best seed choices depend strongly on the population distribution. Here a coordination problem arises: in order to reach equilibrium, different portions of the population must choose different seeds. The task therefore now

[3] We estimate $I_G(x, y)$ using 1000 offline simulations.

[4] Since players are scored against the population distribution, the equilibrium concept here is actually that of an Evolutionary Stable Strategy (ESS)(Maynard Smith 1982).

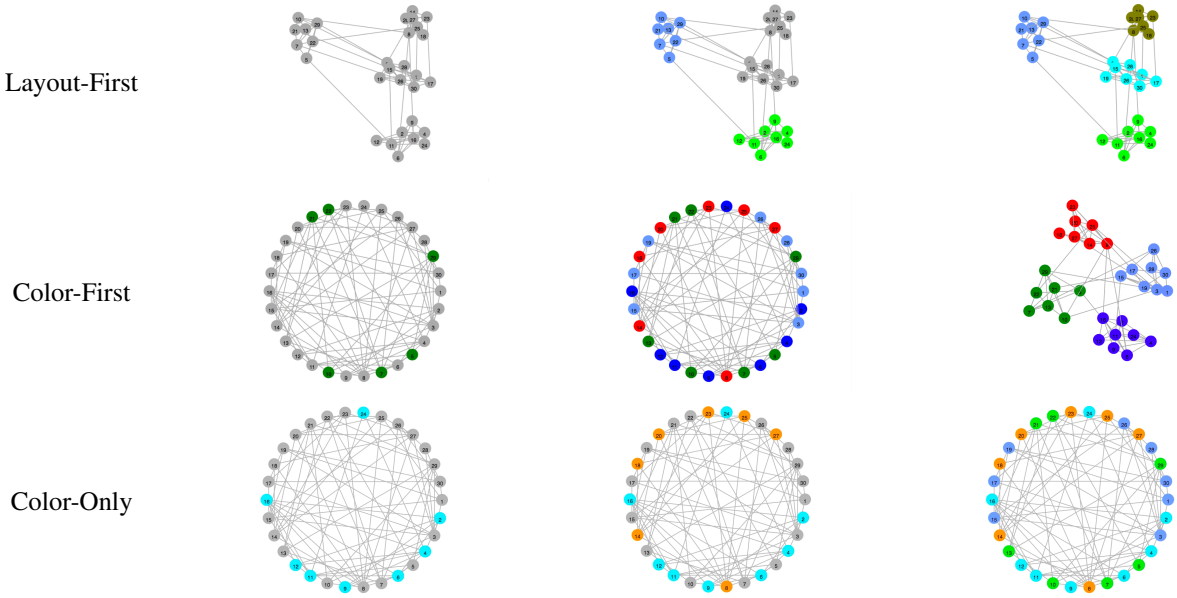


Table 4: Illustration of the layout-first, color-first, and color-only clustering styles. We identify the two players with the highest and lowest style values (0.44 and -0.44, respectively) and also a player for whom this value is undefined, and show snapshots on their play on a particular graph. The layout-first player rearranged the vertices into clusters before assigning colors, whereas the color-first player first completed the assignment and then verified visually. The color-only player made no layout changes.

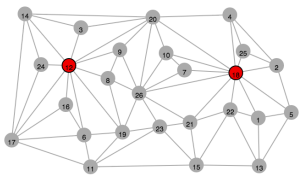


Figure 2: A graph with a dominant strategy, i.e. a seed pair that always infects at least as many vertices as its opponent. The regret on this graph is low (0.0053), because most players converged on the dominant seed choice (shown in red).

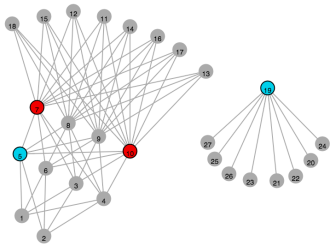


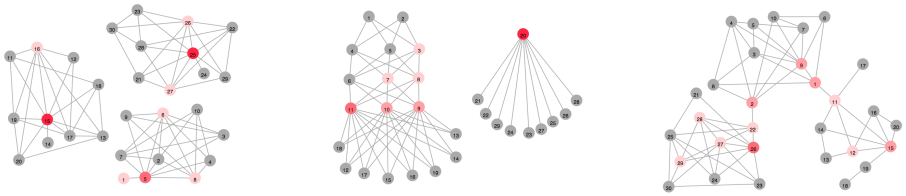
Figure 3: This graph lacks a dominant strategy, and as a result, the population suffered high regret (0.0554). We plot the most popular seed choice in red, and a less popular but higher-scoring alternative in blue.

involves game-theoretic strategizing rather than pure optimization. See Figure 3 for an example.

In general, the population performed much better (i.e. came closer to reaching equilibrium) on graphs with a dominant strategy. The average regret on the subset of 17 graphs for which we know a dominant strategy exists is 0.0068, only about half of the overall average of 0.0129. Furthermore, the correlation the existence of a dominant strategy and the regret of the population across all graphs is -0.48 ($p \approx 0.001$). Both results indicate a strong inverse relationship between the existence of a dominant strategy and performance. Essentially, individual optimization is easy for the subjects, but coordination with other players is difficult.

As a further example, note that all three-component graphs lack a dominant strategy because a player is only allowed two seeds, and so the choice of which components in which to play depends on the population distribution. For instance, if all opponents are playing in components 1 and 2, a player is better off placing one seed in component 3, and therefore infecting that entire component. In order to reach equilibrium, certain fractions of the population have to play in each component pair, but we did not observe such strategic coordination in our experiment. Usually the majority of the population played in two components, and a minority in the third, with the latter subjects enjoying higher scores. As a result, the average regret on three-component graphs is 0.0297, more than twice the overall average.

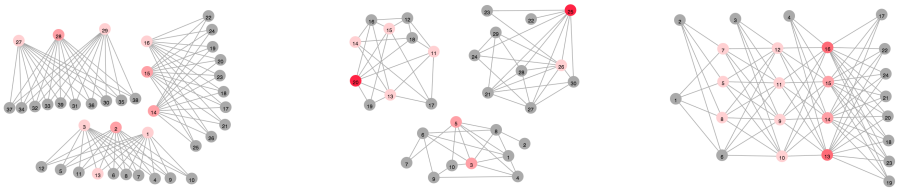
The discrepancy in performance on graphs of each type suggests that when equilibrium requires diversity among strategies, subjects perform more poorly. To quantify this further, we calculate the entropy of the population distribution on each graph. Note that on graphs with a dominant



(a) Regret: 0.0207
Entropy: 2.3953

(b) Regret: 0.0180
Entropy: 2.7181

(c) Regret: 0.0269
Entropy: 3.3467



(d) Regret: 0.0162
Entropy: 4.2272

(e) Regret: 0.0306
Entropy: 2.8857

(f) Regret: 0.0135
Entropy: 3.1386

Figure 4: The graphs shown above all have high population entropy as well as regret. The color saturation of a vertex indicates whether it was chosen as a seed choice by any member of the population, and if so, how popular of a choice it was.

strategy, equilibrium is reached when everyone makes the same seed choice, and therefore the entropy of the distribution is zero at equilibrium (a *pure strategy* equilibrium). But on all other graphs, equilibrium requires a distribution with non-zero entropy (a *mixed strategy* equilibrium). Thus, it is necessary for the population to have higher entropy on these graphs in order to minimize regret. We found that although the population did have higher entropy on these graphs, the population *also* had higher regret. In particular, the correlation between the entropy of the population distribution and the regret on a graph is 0.67 ($p < 10^{-6}$). In other words, the greater the diversity of seed submissions made by the population, the worse the average performance. Thus, it seems that although the population "tried" to achieve equilibrium (as indicated by the high entropy), the players struggled with the coordination of diverse strategies. See Figure 4 for an illustration of the population distribution on a set of graphs that have both high entropy and regret.

Comparisons to Algorithmic Benchmarks

The heuristic against which we compare subject performance is a simple degree-greedy strategy. On a connected graph, this heuristic first chooses the node of highest degree, excludes all of this node's neighbors from consideration, and then chooses the node of next highest degree. If the graph consists of multiple components, the heuristic chooses the highest degree vertex in each of two (randomly selected) components. Ties in degree are broken arbitrarily.

We compare the score achieved by the heuristic's seed choice with the maximum score achieved by any member of the population on each graph. In general, the heuristic makes quite reasonable choices. Thus, it is impressive that on 31 of the 42 graphs, the best seeds found by the population outperform those chosen by heuristic. On the remaining 11 graphs, the two seed choices (and therefore scores) are the same. To explain this discrepancy, we first note that there are a few graphs on which the heuristic makes clearly suboptimal choices (such as near-regular graphs). On some other graphs, the heuristic is handicapped by its inability to choose two adjacent nodes. Most of the time, the heuristic's underperformance is due to the fact that it cannot observe or respond to the population distribution. For instance, on a three component graph, the heuristic has no way of knowing which components the majority of opponents are playing in, and therefore cannot determine a best response.

Individual Strategies

The best performing players were those who observed and understood the strategic tensions outlined above. The third best player commented: *"The first goal is to determine whether or not there are any pure strategy Nash equilibria ... The next step is to determine / gain a sense of what are the current seed choices that are popular ... Lastly for mixed Nash equilibria, based on my understanding of the current popular seed choice, I will select my seed strategy to specifically counter that."* Another in the top five remarked on the difficulty of mixed strategy equilibria: *"In graphs that had connected components ... I saw the most diversity in terms of seed choice ... I also saw that many of these strategies won against some and lost against others, in a kind of cyclical pattern similar to rock-paper-scissors."* In general, players who performed more game-theoretic reasoning achieved higher scores, as evidenced by the positive correlations between both the score of a player and his average number of simulations and "play random opponent" clicks (0.22, $p \approx 0.025$ for the number of simulations, and 0.34, $p < 0.001$ for uses of the play random opponent feature).

Conclusions and Future Work

The results presented here suggest that human subjects are capable of solving many different graph problems of global information. In our experiments, subjects adopted a variety of playing styles, many with a high degree of success. Our findings span three different problem types — search, optimization, and strategy — and are consistent across several different populations. Our work adds to a sparse literature on human subject experiments on whole-network computation and cognition.

Between earlier experiments on collective problem-solving by locally embedded subjects, and our work on individual, whole-network computation, the groundwork is laid for experiments that mix the "in" and "over" approaches. Depending on the task, the optimal organization for a group of subjects or workers might involve the assignment of some parts of a graph to individuals and other parts to groups. We leave the investigation of such richer hybrid decompositions as an intriguing direction for future work.

References

Banerjee, A.; Chandrasekhar, A. G.; Duflo, E.; and Jackson, M. O. 2014. Gossip: Identifying Central Individuals in a Social Network. *ArXiv e-prints*.

Barabási, A.-L., and Albert, R. 1999. Emergence of scaling in random networks. *Science* 286(5439):509–512.

Bharathi, S.; Kempe, D.; and Salek, M. 2007. Competitive influence maximization in social networks. In *Proceedings of the 3rd International Conference on Internet and Network Economics*, WINE '07, 306–311. Springer-Verlag.

Borgatti, S.; Everett, M.; and Freeman, L. 1999. *UCINET 6.0 Version 1.00*. Natick: Analytic Technologies.

Borodin, A.; Filmus, Y.; and Oren, J. 2010. Threshold models for competitive influence in social networks. In *Proceedings of the 6th International Conference on Internet and Network Economics*, WINE '10, 539–550. Springer-Verlag.

Centola, D. 2010. The spread of behavior in an online social network experiment. *Science* 329(5996):1194–1197.

Chasparis, G. C., and Shamma, J. 2010. Control of preferences in social networks. In *Proceedings of the 49th IEEE Conference on Decision and Control*, 6651–6656.

Christakis, N. A., and Fowler, J. H. 2007. The spread of obesity in a large social network over 32 years. *New England journal of medicine* 357(4):370–379.

Cooper, S.; Khatib, F.; Treuille, A.; Barbero, J.; Lee, J.; Beenen, M.; Leaver-Fay, A.; Baker, D.; and Popović, Z. 2010. Predicting protein structures with a multiplayer online game. *Nature* 466:756–760.

Dessi, R.; Gallo, E.; and Goyal, S. 2012. Network cognition. *CEPR Discussion Paper No. DP8732*.

Fowler, J. H.; Christakis, N. A.; Steptoe; and Roux, D. 2009. Dynamic spread of happiness in a large social network: Longitudinal analysis of the framingham heart study social network. *British Medical Journal* 338(7685):23–27.

Gilbert, E. N. 1959. Random graphs. *The Annals of Mathematical Statistics* 30(4):1141–1144.

Giles, M. 2010. A world of connections. *The Economist*.

Goyal, S., and Kearns, M. 2012. Competitive contagion in networks. In *Proceedings of the 44th Annual ACM Symposium on Theory of Computing*, STOC '12, 759–774. ACM.

Kearns, M.; Suri, S.; and Montfort, N. 2006. An experimental study of the coloring problem on human subject networks. *Science* 313(5788).

Kearns, M. 2012. Experiments in social computation. *Communications of the ACM* 55(10):56–67.

Kilduff, M., and Krackhardt, D. 2008. *Interpersonal Networks in Organizations*. Cambridge University Press.

Kirkpatrick, S.; Gelatt, C. D.; and Vecchi, M. P. 1983. Optimization by simulated annealing. *Science* 220(4598):671–680.

Koblin, A. M. 2009. The sheep market. In *Proceedings of the 7th ACM Conference on Creativity and Cognition*, 451–452.

Kunegis, J. 2013. Konect – the Koblenz network collection. In *Proceedings of the International Web Observatory Workshop*, 1343–1350.

Mao, A.; Parkes, D. C.; Procaccia, A. D.; and Zhang, H. 2011. Human computation and multiagent systems: an algorithmic perspective. In *Proceedings of the 25th AAAI conference on artificial intelligence*.

Maynard Smith, J. 1982. *Evolution and the Theory of Games*. Cambridge University Press.

Newman, M. E. J., and Girvan, M. 2004. Finding and evaluating community structure in networks. *Physical Review E* 69(2):026113.

Newman, M. E. J. 2004. Fast algorithm for detecting community structure in networks. *Physical Review E* 69(6):066133.

Newman, M. E. J. 2006. Finding community structure in networks using the eigenvectors of matrices. *Physical Review E* 74:036104.

Suen, H. 2014. Peer assessment for massive open online courses (moocs). *The International Review of Research in Open and Distributed Learning* 15(3).

Suri, S., and Watts, D. J. 2011. Cooperation and contagion in web-based, networked public goods experiments. *PLoS ONE* 6(3):e16836.

Watts, D., and Strogatz, S. 1998. Collective dynamics of 'small-world' networks. *Nature* 393:440–442.

Yuen, M.-C.; King, I.; and Leung, K.-S. 2011. A survey of crowdsourcing systems. In *Proceedings of the 3rd International IEEE Conference on Social Computing*, 766–773.