Do Crowds have the Wisdom to Self-Organize?

Andrea Blasco*
Harvard University

Kevin J. Boudreau
London Business School

Karim R. Lakhani
Harvard Business School

Michael Menietti
Harvard University

Christoph Riedl
Northeastern University

December 2, 2013

Abstract

The "self-organizing" of online crowds—or workers, more generally—into teams is a non-trivial problem of coordination and matching, in a context in which other parties are simultaneously competing for partners. Here, we experimentally investigate the capacity for workers in online crowds to self-organize into teams, within a scientific crowdsourcing contest. We compare matching outcomes and performance to those in a comparison group in which we eliminate the coordination and matching problem altogether (by directly assigning individuals to Pareto efficient teams). Online crowd members do remarkably well relative to the benchmark achieving 13% more functioning teams. Teams also tended to be more effective, by several measures. (We found no evidence these levels

*Email address: ablasco@fas.harvard.edu;
This version is preliminary and incomplete.

depending on the size of the self-organizing pool of workers.) Conditional on having formed, the self-organizing teams also benefit from several advantages in performance.

Introduction

Teams and collaborations are increasingly important in the economy and society, as means of harnessing diverse and complementary skills [9, 14, 2]. At the same time, teams and other organizations are increasingly short-lived, leading workers to act as "free agents" to a greater degree, often moving from team to team, project to project, company to company or among other sorts of organizations[1]. Whereas we have longstanding views of autonomous market coordination—the Smith's "invisible hand" applied to the matching domain, the formation and maintenance of coordinated human activity has long been understood to be province of explicit top-down managerial intervention—the "visible hand" of [4]. Here, we investigate the capacity for people to self-organize into productive teams, exploiting a field experimental context of online innovation contests, i.e. "crowdsourcing", directed to solve challenging computational-algorithmic problems.

Innovation contests are an age old institution for addressing vexing innovation problems in science, technology, and design facing organizations and the society as a whole [3, 5]. In the past decade, this form and approach to organizing innovation activities has gained momentum and prominence, as information technology and digitization of design has allowed contests to be carried out on online "crowdsourcing" platforms in a continuous stream of problems, drawing on a global pool of solvers in domains such as software and algorithm design (e.g., TopCoder and Kaggle), apparel design (e.g., Threadless), electrical engineering (e.g., CADCrowd), mechanical engineering (e.g., LocalMotors), scientific discplines (e.g., Innocentive), financial engineering (e.g., Kickstarter), and other domains. In relation to scientific research, in particular, Crowdsourcing con-

[1]See [13] for a rich exploration of the new forms of contemporary collaboration.

tests have recently been demonstrated to be effective in addressing challenging problems in computational biology and genomics [8, 7]. Crowdsourcing contests can have certain advantages, such as providing high incentives for contestants to exert effort, under certain conditions, and—crucially—in exposing problems to vast pools of diverse individuals and in promoting independent experimentation to encourage a wide diversity of solutions and approaches. To foster these advantages, the majority of crowdsourcing contests today pit individual competitors against other individuals, working autonomously. At the same time, this approach of individual competition fails to combine the resources and knowledge of multiple individuals in teams—potentially limiting the scope of problems that crowdsourcing contests can successfully address. Indeed, in the several conspicuous examples of crowdsourcing in which teams were allowed to form from the crowd–such as the Netflix prize and in DARPA competitions—extraordinary levels of performance have been achieved. Nonetheless, it is difficult to discern from isolated examples whether it is more generally possible for online crowds to effectively form teams and to do so in some sort of sensible or even predictable fashion, to "self-organize". Can a self-organizing "market for partners" effectively clear on its own, or is there any evidence that it might do so—at least to some extent—absent the visible hand of managerial intervention? The answer is not obvious. Matching is a difficult and complex task: it not only involves finding a suitable match in a context of imperfect information, frictions and coordination problems, but also doing so in a context in which everyone else is at the same time searching (and competing) for partners. Gaining empirical evidence on these issues will help us understand the limits of crowdsourcing contests as an organizational approach to solving innovation problems, in particular, and also shed light on the limits of self-organization more generally.

The Experiment

To investigate these questions, we conducted an experiment. We compared the matching into teams and consequent performance of self-organizing pools of individuals to a benchmark group in which the problem of coordinating matches was eliminated by applying an assignment procedure geared to implementing matches in a precise way, described below. The experiment was run on a leading online crowdsourcing platform, TopCoder, and the experiment involved a total of 462 participants competing within a four-week contest sponsored by the United States Patent and Trademark Office (USPTO) to develop advanced algorithms to identify technical illustrations labels and captions from drawing pages of US patent documents.

A total of 926 individuals were initially drawn from the pool of TopCoder members on the basis of our advertising (see the Appendix). In this initial communication to attract participants we broadly characterized the nature of the problem[2] and prizes. It was also communicated that participants would be forming teams of two to compete in the challenge and that they would not be working with people with whom they had established ties. Those signing up to the contest completed a registration questionnaire to describe themselves, their experience and preferences for alternative programming language.

At the outset of the experiment, participants were simultaneously informed of the details of the exercise: the precise nature of the task to be performed, and were given a seven-page description of the problem, the rules of the rank-order contest, how it would be scored, nature of prizes, etc. (See the website `community.topcoder.com/ntl/uspto/` for more details). Broadly, they would be divided into randomly assigned groups of 40, in which they would form teams of two, within which they would complete to develop solutions to the problem. The development or production phase of the contest then took place over 31 days. Each team could submit as many solutions as they wished, as this provided a means of assessing the quality of their algorithms, as each algorithm was automatically scored by the platform according to an automated test suite. In each independent group, prizes were awarded according to the final ranking of the score obtained for the last proposed solution. In particular, the overall

[2] Including mention of the necessary IT skills to address it, some basic description of the dataset they would be working on, and the timing of the competition.

top two teams were awarded $10,000 and $5,000 respectively. Only after this communication, those 926 signing up were randomly assigned to 22 groups of 40 people (plus a 23^{th} residual group of 46).

As a basis for developing our treatment groups, we then proceeded to collect the teammate preferences of all subjects. Specifically, they were instructed to rank order all other prospective partners assigned to the same random group of 40. Thus, they were each asked to rank order 39 names (or, specifically, online "handles"). The order in which names appeared was randomized in each instance. Profiles included information from the sign-up questionnaire and information from past experience on the TopCoder platform: demographics (age, gender, education, geographic origins), self-assessed competence in various fields (command of english, programming language, relevant IT skills) and TopCoder skill rating measure based on past performance[3].

Of the original 926 individuals who signed up, a total of 462 (50%) of subjects completed the rankings.[4] This set of 462 individuals were then taken as subjects in the experiment and were generally higher skilled and had more years of programming experience relative to those who chose not to continue. Regression of rank orders on the characteristics of individuals in each list, as in [1], suggests preferences as might be expected in relation to increasing ex ante expectations

[3]Subjects had at least two objective (i.e., non self reported) measures of relative ability of partners. Each measure was the result of individual past performance in two very popular types of algorithmic contest – Single Round Matches (SRM) and Marathon Matches (MM) – that are run on a monthly basis by TopCoder. The two measures are highly correlated (i.e., $\rho = .57$). In our sample there are more competitors with experience in SRM (52%) than in MM (25%). For the purpose of our analysis, we will focus on the MM measure, simply because the competition resembles more closely s MM than an ALGO contest (see the TopCoder website for further information). However, results are qualitatively similar if we use either the ALGO measure, or an index defined as a weighted average of the two measures.

[4]We find no systematic differences across treatments resulting from this deliberate attrition process, with 227 or 52% in the Self-Organizing and 235 or 48% in the Benchmark Group treatment. Further, each list of prospective partners received a median of 12 edits before being returned as ranking of preferences (see Figure 1) and the number of edits does not appear to change systematically across groups or individuals.

of productivity of teams. Individuals had strong taste for partners of high skill rating with some degree of recognizing individuals more particular traits such as coming from the same country, same programming language, and other characteristics. Importantly, dropouts to the originally 23 randomly assigned groups of 40 also produced random variation in the number of subjects in each group, with a mean of 20 in each group, ranging from a minimum of 15 to a maximum of 26. We will later exploit this variation in the "size of crowds" for our analysis. Preferences were also then used to generate both Self-Organizing treatments and a Benchmark comparison treatment, as will be described below.

The Self-Organizing Group and the ("Pareto Efficient") Benchmark Comparison Group It is at this point that the 23 groups were themselves randomly divided into Self-Organizing treatments (11 groups) and Benchmark comparison treatments (12 groups). (See Figure 2 for the procedural summary of the Self-Organizing and Benchmark treatments). In Self-Organizing, we set a deadline of five days by which they had to bilaterally form teams with people of the same pool. To aid in team formation we provided an instant messaging tool and a web forum, thereby we could track communication patterns during the process of matching. Furthermore, we established that as soon as a pair agreed to form a team together that decision could not be revised. All teams that formed entered into the competition and all individuals failing to form teams were excluded from the sample. This led to still more variation in the number of competing teams that appeared in each independent group.

We then treated the 12 remaining groups as our benchmark to which we could compare the Self-Organizing treatment, so as to assess the extent and quality of matching. We construct this benchmark to provide both theoretical, yet also practical empirical meaning as a comparison group, in terms of what "well-coordinated" matching might be judged. Here we rely on economic theory (i.e., [6]) to establish that well-coordinated matches are those from which members will find impossible to rationally choose to deviate from in favor of an alternative willing partner (i.e., matchings are "stable"). The precise algorithm is described in the Appendix and follows a Shapley-Gale benchmark of maximum stable matching [11, 12]. The algorithm essentially takes each subjects stated preferences as inputs to iteratively generate a maximal set of teams from which subjects would not prefer to deviate. In this sense the benchmark represents a "Pareto Efficient" configuration of matches.

For this benchmark to serve as an effective and meaningful comparison group, we do not stop at a theoretically construed Shapley-Gale benchmark. We wish to compare the ability of the Self-Organizing group to an *empirically* relevant benchmark. And so, we run an experimental Shapley-Gale comparison group to allow us to compare the Self-Organizing group to a theoretically-motivated but empirically meaningful benchmark. Thus, a total of 235 participants from 12 matching pools were sorted into teams by an algorithm based on [11], thereby selecting matches from the set of theoretically maximum-stable matchings given the collected individual rankings. This resulted in each subject (i.e., those who submitted partner preferences) receiving a communication just prior the competition with the name of the individual with whom the partnership was established. Therefore, we may think of the Benchmark comparison group as one that faces identical conditions to the Self-Organizing group, except that the coordination problem of finding matches or teams has been eliminated and the remaining choices are simply whether to proceed or not, and then to engage in development.

Results

We first investigate the extent to which Self-Organization groups were able to form teams relative to the those in the Benchmark treatment and then proceed to investigating any performance difference that resulted from the differences in the treatments. Overall, we find that matching and team formation in the Self-Organizing treatment proceeds remarkably well by multiple measures. Furthermore, the process of self-organization, itself, appears to generate certain benefits, as will be described in turn.

The Extent and Quality of Self-Organization A total of 144 individuals formed teams in the Self-Organizing groups, with a mean of 13 individuals forming teams in each group (minimum of 10 and maximum of 16). Therefore, remarkably, the Self-Organizing groups were able to achieve just over 67% of the level of team formation as the comparison group in which the coordination and matching problem was eliminated altogether.

Consistent with our hypothesis of costs of search and effort to coordinate in matches, it appears that those who eventually formed teams exerted considerably more effort in the search process with 93% either sending a chat message (91%) or posted on the web forum (43%); by contrast, less than half (47%) of the unmatched participants actively searched for a partner during the team formation phase, with just 32% sending chat messages and 18% posting on the web forum

Further, it appears that the willingness of individuals to exert the fixed and sunk cost of search in order to form a team and enter the competition is nonrandom. The skills of those forming teams in Self-Organization are higher than those forming teams in the Benchmark treatment (i.e., 40% higher average skill rating). Those teams choosing to actively participate in the Benchmark treatment are also higher skilled than those who choose not to enter. Both patterns are consistent with positive sorting on skills in a rank-order contest as in [10], with the fixed cost of sorting accentuating this pattern. Therefore, although search costs may impose certain costs of coordination, they at the same time appear to also generate positive sorting on skills in this context of a rank order contest.

Apart from the overall level of matches, we next consider the nature and quality of matches formed. A direct indication of the quality of matching is to simply compare the rank that partners assigned one another prior to forming a team. So it turns out that partners in Self-Organizing groups were ranked higher on average than partners in the comparison group, with on average 10.6 ranks against 8.4 ranks.

Consistent with greater search frictions, individuals forming teams in the

Self-Organized treatment generally joined with a slightly less preferred teammate, according to their earlier-expressed preferences, with a difference of 2.24 ranks (standard deviation of 0.95) which is statistically significant at 5% level in relation to matches in the Benchmark treatment. While this is lower, the difference might be regarded as remarkably small (out of the 39 other possibly ranked individuals). Further, as seen in Figure 4, it appears the Self-Organizing treatment has minimal effect on highest-skilled partners, where the difference in the quality of matching across Self-Organizing and Benchmark is minimal and not statistically significant.

Apart from the raw comparison of rank order preferences formed to teams, at least as meaningful measure of the quality of matches is the number of pairs of participants that have not matched among each other when would both prefer to do so. These are referred to in the literature as "blocking pairs"—as these coalitions "block" the achieved matching because other preferred matches can form and the matching is said "unstable"—but a more formal definition can be find in [6]. In our case, the number of *blocking pairs* can be assessed empirically from the matchings and the collected rankings. So we define an empirical counterpart of a blocking pair as a $(0,1)$ variable which scores one whenever, for any pair of agents (i,j), given that i and j are matched with k and z respectively, the following condition is verified:

$$\mathbf{1}\left(rank_i(k) - rank_i(j) \geq p\right) * \mathbf{1}\left(rank_j(z) - rank_j(i) \geq p\right) > 0 \qquad (1)$$

where the function $\mathbf{1}()$ is an indicator function, and p is a fixed difference in the observed ranks, which allows us to take into account the presence of errors in the collected rankings.

Figure 3 shows that if we remove from the sample all agents who ended up without a partner and we fix p = 1, we obtain an average of 0.96 blocking pairs per person, as compared to .14 blocking pairs per person calculated on those who entered the competition within the Benchmark treatment. Therefore, the Self-Organizing treatment implements roughly the second best possible team (i.e., literally one lower rank than what might be considered best), on average, relative to the originally-stated preferences. Further, the difference remains strictly positive for all levels of $p \leq 8$. This evidence strongly rejects the absence of frictions in the market and might suggest that a self-organizing crowd is doing worse in terms of efficiency compared to the benchmark.

We also investigate the extent to which these results might depend upon the size of the crowd. For example, as a crowd grows larger, there might be on the one hand more frictions of searching. On the other hand, there might be fewer "small numbers" matching problems as there are more options for partners available. To investigate any possible effects, we exploit here the variation in numbers of subjects in the matching pool in each of the 11 Self-Organizing groups, from the number of teams formed to the average rank of a partner. Table 4 shows we find no systematic differences when regressing any of our measures of the extent or quality of teams formed (team formation, rank of partner, number of blocking pairs), on the "size of the crowd" or number of subjects in each pool. We find no effects with or without control variables. Therefore, we find no evidence that Self-Organization either deteriorates or improves with the size of the crowd.

Team Performance Implications We next measure the extent to which differences in matching between the two treatments result in performance differences, once teams are organized. The earlier results document only marginal deterioration in the extent and quality of matching in the Self-Organizing treatment relative to that in the Benchmark treatment, and in relation to self-reported preferences, in which the matching and coordination task is eliminated altogether. From this perspective, we might then expect some marginal diminution in team performance inasmuch as matching is in some sense "suboptimal" or hindered by coordination and matching frictions. At the same time, the process of Self-Organization might itself also generate offsetting benefits, if the process generated additional information about partners or otherwise the process of self-organization imbued teams with greater productivity through socialization processes. Overall, we find here that any marginally negative effects of self-organizing are more than offset by compensating benefits of the process of self-organizing that translate to performance.

Whereas the number of implemented team matches was necessarily lower in Self-Organizing group relative to the Benchmark Group, as earlier, there were in fact a greater number of teams that became "active", in the sense of making at least one submission: in the Self-Organizing Group 34 teams versus just 30 teams in the Benchmark treatment. In proportional terms, this implies 47% of formed teams became active under Self-Organization versus 25% in cases in which the coordination and team-formation problem was entirely eliminated. And Figure 5 shows that most of the difference is due to lower-skilled teams.

Differences in levels of activity are even wider when considering the overall level of activity, in terms of the number of submissions of solutions. The 72 teams that formed in the Self-Organizing pool generated a total of 607 solution submissions, whereas the 115 teams in the Benchmark treatment in all, whereas only 347 submissions came from teams of the Benchmark Group.

Turning to the quality of the final solutions developed under either treatment, we compare the precise measures of problem-solving performance that were based on subjecting each solution to an automated test suite. If we compare the distribution of final scores achieved in each regime, the median and mean differences are neither statistically nor substantively different from one another across either regime, and a Kolmogorv-Smirnov test do not reject the null hypothesis that the scores were in fact drawn from the same distribution. Yet, the "right tail" of the distribution of scores appears to be longer in the Self-Organizing treatment. However, simulating confidence intervals with bootstrapping fails to find these differences to be statistically different.

Conclusions

In this paper we investigated the capacity for a crowds to self-organize, to effectively form teams, or to clear the market for team partners. We developed a controlled experimental comparison in which we compared self-organizing groups to a theoretically motivated benchmark treatment, based on a principle Pareto efficient teams. The benchmark is therefore, in a sense, something of an ideal case in which the coordination and matching problem is removed altogether.

Although we find evidence of search and matching frictions and costs and a diminution of the extent and quality of matching under self-organization in relation to an ideal benchmark, what is most remarkable is this diminution appears marginal. Self-organization implemented roughly two-thirds of the number of teams that were implemented by the ideal benchmark. Moreover, the costs imposed by search and matching had the effect of generating more positive sorting on skills. Further, the extent and quality of matching were best among highest skilled subjects, indicating still smaller costs of self-organizing. We found no evidence that the quality and extent of matching varied within the variation in the size of crowd studied here.

Apart from these results indicating some considerable capacity for the crowd in this instance to self-organize into teams, with only marginal costs, examination of behavior and performance of teams once formed indicates that the process of self-organizing generates compensating benefits. We found no evidence that performance in the self-organizing group was any lower than in the ideal benchmark group. To the contrary, self-organizing teams were more likely to deliver a working solution, to deliver more solutions, to deliver solutions of higher quality (although the difference is not statistically significant).

References

[1] Paul D Allison and Nicholas A Christakis. Logit models for sets of ranked items. *Sociological methodology*, 24(1994):199–228, 1994.

[2] Nicholas Bloom and John Van Reenen. Human resource management and productivity. *Handbook of labor economics*, 4:1697–1767, 2011.

[3] Liam Brunt, Josh Lerner, and Tom Nicholas. Inducement prizes and innovation. *The Journal of Industrial Economics*, 60(4):657–696, 2012.

[4] Alfred D Chandler. *The visible hand: The managerial revolution in American business*. Harvard University Press, 1977.

[5] Lee Davis and Jerome Davis. How effective are prizes as incentives to innovation? evidence from three 20th century contests. In *DRUID summer conference*, volume 2004, 2004.

[6] David Gale and Lloyd S Shapley. College admissions and the stability of marriage. *The American Mathematical Monthly*, 69(1):9–15, 1962.

[7] Benjamin M Good and Andrew I Su. Crowdsourcing for bioinformatics. *arXiv preprint arXiv:1302.6667*, 2013.

[8] Karim R Lakhani, Kevin J Boudreau, Po-Ru Loh, Lars Backstrom, Carliss Baldwin, Eric Lonstein, Mike Lydon, Alan MacCormack, Ramy A Arnaout, and Eva C Guinan. Prize-based contests can provide solutions to computational biology problems. *Nature biotechnology*, 31(2):108–111, 2013.

[9] Edward P Lazear. Globalisation and the market for team-mates. *The Economic Journal*, 109(454):15–40, 1999.

[10] Benny Moldovanu and Aner Sela. Contest architecture. *Journal of Economic Theory*, 126(1):70–96, 2006.

[11] Jimmy JM Tan. A maximum stable matching for the roommates problem. *BIT Numerical Mathematics*, 30(4):631–640, 1990.

[12] Jimmy JM Tan. A necessary and sufficient condition for the existence of a complete stable matching. *Journal of Algorithms*, 12(1):154–178, 1991.

[13] Ruth Wageman, Heidi Gardner, and Mark Mortensen. The changing ecology of teams: New directions for teams research. *Journal of Organizational Behavior*, 33(3):301–315, 2012.

[14] Stefan Wuchty, Benjamin F Jones, and Brian Uzzi. The increasing dominance of teams in production of knowledge. *Science*, 316(5827):1036–1039, 2007.

1 Appendices

1.1 Tables and Figures

Table 1: Summary Statistics Self-Organizing Group

Statistic	N	Mean	St. Dev.	Min	Max
groupID	227	5.982	3.137	1	11
ismale	227	0.938	0.241	0	1
birthyear	227	1,984.643	8.785	1,900	1,993
timezone	227	2.460	5.195	−8.000	10.000
avgRank	227	18.919	4.167	6.600	29.000
avgEdits	224	11.574	3.232	0.564	19.487
mm_rating	227	541.551	716.960	0	2,591
mm_num	227	3.445	7.599	0	59
algo_rating	227	777.524	712.634	0	2,873
algo_num	227	23.273	40.457	0	285
country_originOther	227	0.467	0.500	0	1
country_originChina	227	0.194	0.396	0	1
country_originIndia	227	0.172	0.378	0	1
country_originUnited.States	227	0.167	0.374	0	1
educationCollege.degree	227	0.436	0.497	0	1
educationDoctoral.degree	227	0.031	0.173	0	1
educationMaster.degree	227	0.366	0.483	0	1
educationSecondary.school	227	0.167	0.374	0	1
prefSpokenLangCantonese	227	0.013	0.114	0	1
prefSpokenLangEnglish	227	0.762	0.427	0	1
prefSpokenLangFrench	227	0.009	0.094	0	1
prefSpokenLangGerman	227	0.004	0.066	0	1
prefSpokenLangHindi	227	0.018	0.132	0	1
prefSpokenLangMandarin	227	0.079	0.271	0	1
prefSpokenLangPolish	227	0.013	0.114	0	1
prefSpokenLangRussian	227	0.053	0.224	0	1
prefSpokenLangSpanish	227	0.044	0.206	0	1
prefSpokenLangTamil	227	0.004	0.066	0	1
prefProgLangC.	227	0.106	0.308	0	1
prefProgLangC..	227	0.471	0.500	0	1
prefProgLangJava	227	0.326	0.470	0	1
prefProgLangPython	227	0.093	0.290	0	1
prefProgLangVB	227	0.004	0.066	0	1

Table 2: Summary Statistics Benchmark Group

Statistic	N	Mean	St. Dev.	Min	Max
groupID	235	17.617	3.444	12	23
ismale	235	0.966	0.182	0	1
birthyear	235	1,984.098	9.669	1,900	1,993
timezone	235	2.102	5.259	−8.000	10.000
avgRank	235	19.541	3.912	8.920	30.188
avgEdits	228	11.629	3.517	0.051	19.487
mm_rating	235	471.643	678.679	0	2,821
mm_num	235	4.021	9.197	0	71
algo_rating	235	766.570	705.937	0	2,911
algo_num	235	27.464	48.190	0	299
country_originOther	235	0.447	0.498	0	1
country_originChina	235	0.157	0.365	0	1
country_originIndia	235	0.187	0.391	0	1
country_originUnited.States	235	0.209	0.407	0	1
educationCollege.degree	235	0.477	0.501	0	1
educationDoctoral.degree	235	0.055	0.229	0	1
educationMaster.degree	235	0.323	0.469	0	1
educationSecondary.school	235	0.145	0.353	0	1
prefSpokenLangCantonese	235	0.026	0.158	0	1
prefSpokenLangEnglish	235	0.770	0.422	0	1
prefSpokenLangFrench	235	0.030	0.170	0	1
prefSpokenLangGerman	235	0.000	0.000	0	0
prefSpokenLangHindi	235	0.013	0.113	0	1
prefSpokenLangMandarin	235	0.068	0.252	0	1
prefSpokenLangPolish	235	0.017	0.130	0	1
prefSpokenLangRussian	235	0.068	0.252	0	1
prefSpokenLangSpanish	235	0.009	0.092	0	1
prefSpokenLangTamil	235	0.000	0.000	0	0
prefProgLangC.	235	0.102	0.303	0	1
prefProgLangC..	235	0.511	0.501	0	1
prefProgLangJava	235	0.306	0.462	0	1
prefProgLangPython	235	0.068	0.252	0	1
prefProgLangVB	235	0.013	0.113	0	1

	Matched	Unmatched	Total
Non-sender	13	56	69
Sender	131	27	158
Non-recipient	13	21	34
Recipient	131	62	193
Non-posting	82	68	150
Posting	62	15	77
Active	134	39	173
Non-active	10	44	54

Table 3: Expending effort in matching

Table 4: Variation of performance with respect to group size

	Dependent variable:				
	finalScore.max	finalScore.mean	submissions	matched.sf	rank_partner.mean
	(1)	(2)	(3)	(4)	(5)
Group size	−1.539	−0.841	−0.047	0.390	−0.008
	(3.267)	(1.894)	(0.079)	(0.213)	(0.208)
Self-Org = 1	30.404	13.136	0.349		2.064
	(19.526)	(11.322)	(0.472)		(1.242)
Odd number = 1	−19.400	−7.617	−0.490	−0.421	0.389
	(20.958)	(12.153)	(0.507)	(1.178)	(1.333)
Constant	100.943	59.395	3.837**	5.149	8.720*
	(67.896)	(39.371)	(1.642)	(4.468)	(4.320)
Observations	22	22	22	11	22
R^2	0.167	0.098	0.090	0.310	0.134

Note: *p<0.1; **p<0.05; ***p<0.01

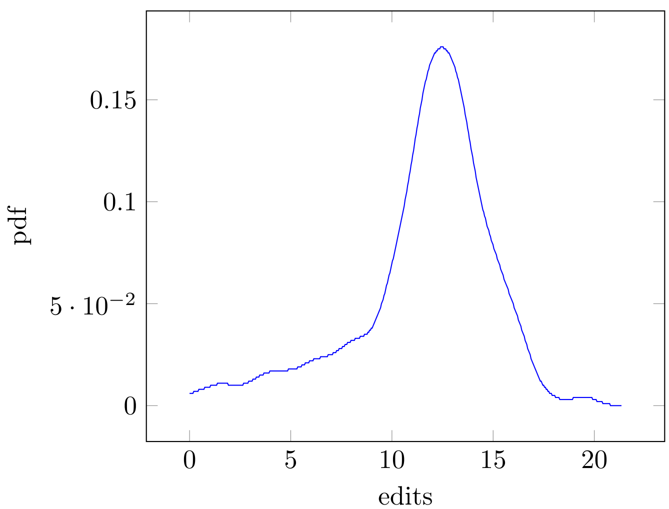

Figure 1: This figure shows the density of the variable *edits* per subject. Per each subject, the variable *edits* is computed as the average of the (absolute value) difference between the initial position of each alternative in the list and the returned final rank of each alternative. Higher values of edits imply higher dissimilarity between the initial random ordering of alternatives and the returned rankings of preferred partners.

Figure 2: Timeline of the Experiment

Self-Organizing Group	**Benchmark Group**
1.1 Registration & assignment to groups of 40 people	1.1 Registration & assignment to groups of 40 people
1.2 Rank-order prospective partners	1.2 Rank-order prospective partners
1.3 Five-day market process of team formation	1.3 A maximum-stable matching is implemented in each group and each team member is notified the identity and the contacts of the assigned partner
1.4 Market clears; matched teams are ratified & unmatched subjects are excluded from our sample	1.4 Four-week submission phase (31 days)
1.5 Four-week submission phase (31 days)	

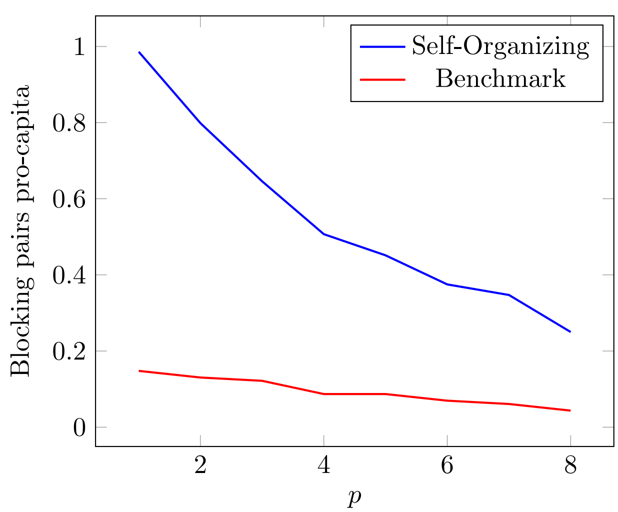

Figure 3: Blocking pairs pro-capita across treatment groups as a function of different values of the threshold p (as defined in the text)

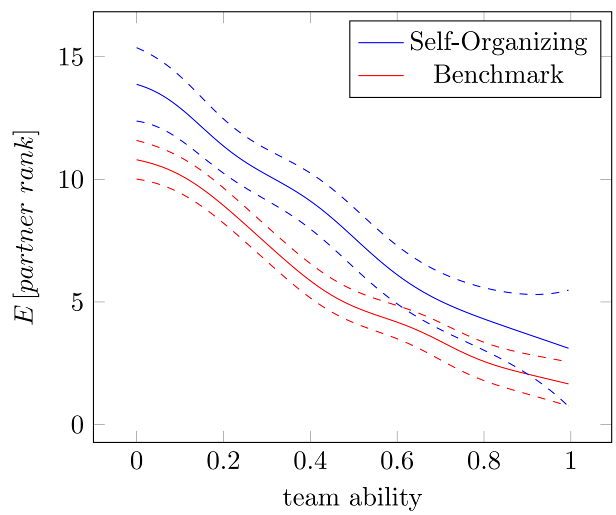

Figure 4: Non-parametric regression results with a sample of $N = 187$ teams. The dependent variable is the average rank that partners assigned one another. The variable of team ability is obtained by the average (MM) skill rating of the team members (normalized on the unit interval).

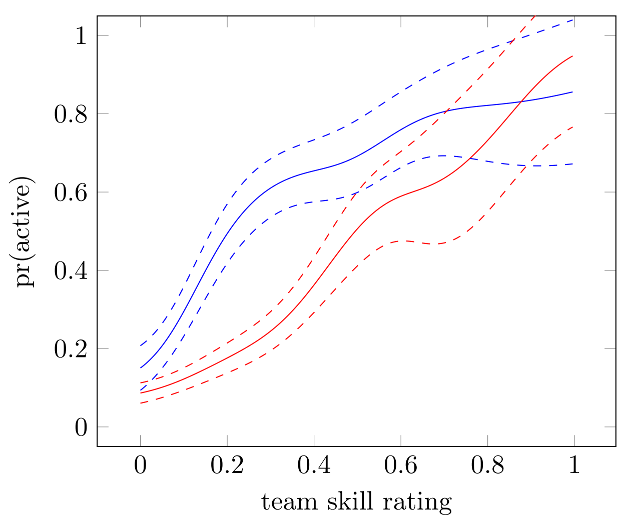

Figure 5: Non-parametric regression results with a sample of $N = 187$ teams. The dependent variable is a variable indicating whether the team submitted at least one working solution (i.e., a solution of positive score). The variable of team ability is obtained by the average (MM) skill rating of the team members (normalized on the unit interval).

1.2 Experimental Documents

1.3 Matching Algorithm

The mediated matching algorithm we implement is based on the algorithm in Tan (1991). It is well-known that two-sided matching problems, like marriage markets, have stable matches between individuals. In a stable match no *blocking pairs* exist. Two people cannot break their current match to form a new match in which each prefers her new partner to her old partner. In one-sided matching problems, like roommate matching, a stable matching may not exist. Tan shows that a more general structure called a stable partition is guaranteed to exist. It is always possible to take a one-sided matching problem and create a stable partition consisting of a set of matches within which no blocking pairs exist, and an odd party. The odd party is a set of individuals who cannot be matched without creating a blocking pair. The set of odd party members cannot be matched under any stable partition.

At a high-level, the mediated matching algorithm progressively assigns matches and removes the matched individuals from the matching problem. It iterates until all individuals have been assigned.

At the start of an iteration the matching problem is split into 1. stable matches, 2. the odd party, 3. and single-matches, if there is at most one individual who cannot be matched because of an odd number of individuals. If the odd party is not empty, then its members are assigned partners through a random serial dictatorship. Using given agent priorities, each member of the odd party is assigned her most preferred partner among the unassigned odd party members in order from highest to lowest priority. The odd party is guaranteed to have an odd number of members. As such, one member of the odd party will be unmatched at the end of the assignment. The assigned odd party members are removed from the matching problem. Then another iteration begins. Once no odd party members remain in the matching problem, the stable matches are assigned. These are identified by the Tan algorithm in the process of splitting the problem. The assigned individuals are then removed from the matching problem and another iteration begins. At this point, at most a single individual remains in the matching problem. She is assigned a special identifier as her partner indicating no possible match and is removed from the matching problem. Finally, all individuals have been assigned and the algorithm ends