In A Flash: Crowdsourcing Organizations, Collaboration, and Research

Michael Bernstein
Stanford University
A personal story
Achieving More Together

Crowdsourcing is a technology for amplifying human effort.
CREATING INFRASTRUCTURES FOR COMPLEX GOALS

Crowdsourcing on-demand groups of experts from Upwork [Retelny et al. 2014]

Creating infrastructures for complex goals

24 HOURS

NAPKIN SKETCH

HI-FI PROTOTYPE
CREATING INFRASTRUCTURES FOR COMPLEX GOALS

Crowdsourcing on-demand groups of experts from Upwork
[Retelny et al. 2014]
CREATING INFRASTRUCTURES FOR COMPLEX GOALS

Crowdsourcing on-demand groups of experts from Upwork
[Retelny et al. 2014]

Celebrate the work of Professor Terry Winograd, and his building a computer in his garage when a child.

SCRIPT IDEA

SCRIPT

STORYBOARD

CHARACTER DESIGN

BACKGROUND DESIGN

ANIMATION

DIRECTOR

ANIMATOR

SOUND ENGINEER

SCRIPTWRITER

ILLUSTRATOR

MUSIC

VOICEOVER

MIX

EDITING

VIDEO TRACK

AUDIO TRACK

ANIMATED VIDEO
CREATING INFRASTRUCTURES FOR COMPLEX GOALS

Crowdsourcing on-demand groups of experts from Upwork [Retelny et al. 2014]

Computational crowdsourcing techniques enable...

- **Modularity**
- Elasticity
- Pipelining
- Automatic creation
CREATING INFRASTRUCTURES FOR COMPLEX GOALS

Crowdsourcing on-demand groups of experts from Upwork [Retelny et al. 2014]

Computational crowdsourcing techniques enable...

- Modularity
- **Elasticity**
- Pipelining
- Automatic creation
CREATING INFRASTRUCTURES FOR COMPLEX GOALS

Crowdsourcing on-demand groups of experts from Upwork [Retelny et al. 2014]

Computational crowdsourcing techniques enable...

- Modularity
- Elasticity
- Pipelining
- Automatic creation
CREATING INFRASTRUCTURES FOR COMPLEX GOALS

Crowdsourcing on-demand groups of experts from Upwork [Retelny et al. 2014]

Computational crowdsourcing techniques enable...

- Modularity
- Elasticity
- Pipelining
- Automatic creation
CREATING INFRASTRUCTURES FOR COMPLEX GOALS

Crowdsourcing on-demand groups of experts from Upwork [Retelny et al. 2014]

Computational crowdsourcing techniques enable...

Modularity
Elasticity
Pipelining

Automatic creation
CREATING INFRASTRUCTURES FOR COMPLEX GOALS

Crowdsourcing on-demand groups of experts from Upwork [Retelny et al. 2014]

Computational crowdsourcing techniques enable...

- Modularity
- Elasticity
- Pipelining
- **Automatic creation**
NARROW VS. OPEN-ENDED GOALS

These teams were restricted in what they can achieve

- Interface iteration, not product design
- Rapid prototypes, not software engineering
- Animating a prompt, not film or game production

Could we achieve open-ended, complex goals such as product design, software development, and game production?
No. We couldn’t create a process structured enough for computation to help, without over-constraining it.
INFRASTRUCTURE: CROWD ALGORITHMS

Crowdsourcing’s infrastructure is based on algorithmic primitives

Modularize and pre-define all actions

This abstraction allows computation to decide which behaviors are taken, when, and by whom; optimize, error-check, combine submissions, and more
The limits of algorithms

Open-ended and complex goals are fundamentally incompatible with a requirement to pre-define all behaviors [Van de Ven, Delbecq, and Koenig 1976; Rittel and Weber 1973; Schön 1984]

This infrastructure confines crowdsourcing to goals so predictable that they can be entirely pre-defined
“Peer production is limited not by the total cost or complexity of a project, but by its modularity, the granularity of its components, and the cost of integration.”

[Benkler 2002]
The very thing that gives crowdsourcing systems their leverage is also preventing them from achieving complex and open-ended outcomes.
An alternative approach: crowds structured not like algorithms, but like organizations
1) Enable crowd collectives to achieve complex and open-ended goals

2) Recruit effective collaborators despite unpredictable availability

3) Crowdsource research itself, providing global access to upward mobility
Flash organizations

Valentine, Retelny, To, Rahmati, Doshi, Bernstein. CHI 2017.
Flash organizations carry out open-ended, complex goals that were previously out of reach for crowdsourcing: product design, software development, and game production.
FOUNDRY

Web platform that supports authoring, reconfiguring, and running flash organizations
CHALLENGES

1) Organizations assume **asset specificity**: people developing effective collaboration patterns over time [Williamson 1976]

...but on-demand crowds do not offer asset specificity
CHALLENGES

1) Organizations assume **asset specificity**: people developing effective collaboration patterns over time [Williamson 1976]

   ...but on-demand crowds do not offer asset specificity

2) Organizational structures require **constant reconfiguration** so that the organization can adapt as it proceeds

   ...but algorithmic models have not allowed for open-ended reconfiguration
APPRAOCH: ROLE STRUCTURES

Inspired by film crews and disaster response teams [Bigley 2001; Bechky 2006; Klein et. al 2006; Valentine & Edmondson 2015]

Role structures enable interaction based on knowledge of roles rather than asset-specific knowledge of each other
COMPUTATIONAL ORGANIZATIONAL STRUCTURES

Roles: parametrize required expertise

Teams: groups of workers with shared goal

Hierarchy: nested roles that determine decision rights
FOUNDORY ROLE STRUCTURES

Map each role onto a skill in the Upwork labor market

Nest roles into teams to indicate hierarchy
ROLE-BASED HIRING + ONBOARDING

On-demand hiring from the labor market

Project: Question and Answer Web Application
Task: Homepage & Login Wireframes
Position in Queue: No. 1
Deadline to Accept Position: 10 minutes

Accept this position
Decline this position

Task Available

Congratulations! You are at No. 1 position. Application project.

Please read the following information carefully before you get to the hiring queue. However, to reinforce again, if you are not sure of the hiring queue, you can read about your task, and start tracking work time. Note that time for reviewing the previous materials, etc. are accounted for as work time.

Pay close attention to the task description, the 'inputs' (what other workers have handed off to you), and the deliverables you are expected to create.

Your Task

This is YOUR task. You can now end this tour, and click on the task rectangle and click start to read about your task, and start tracking work time. Note that time for reviewing the previous materials, etc. are accounted for as work time.

Automated, role-specific onboarding
To enable reconfiguration of the organizational structures: branching and merging inspired by version control.
Any member can branch, edit, and issue pull requests against any organizational structure: roles, teams, hierarchy, tasks.

Pull requests are reviewed up the hierarchy and merged through a three-way diff.
COMPUTATIONAL AFFORDANCES

1) Asset specificity
   Hierarchical role structures
   Rapid hiring and onboarding

2) Adaptation
   Reconfigurable tasks, teams, and hierarchy: top-down and bottom-up
   Branch+merge version control-style reconfiguration
Field study: recruit outside leaders to pursue open-ended goals that have remained out of reach for crowdsourcing

**Leader**

**Medical resident**

**Open-ended goal**

- **EMS Report**
  - Develop prototype application for EMTs to transmit patient information en route to hospital

- **Tech lab employee of a large company**
  - Design and manufacture a storytelling card game with accompanying mobile application

- **Kickstarter team**
  - Develop a workshop planning portal consistent with enterprise standards and branding
End users spun up and led entire organizations in six weeks, convening new workers on-demand within fourteen minutes on average.
2 mobile applications, 3 full-stack web applications in 52,000 lines of code, 2 illustrated card decks
639 tasks, 3261 person-hours of work across 35–46 days from engineers, designers, testers, poets, and others
Passed quality review by neutral experts
EMS TRAUMA REPORT

[Diagram showing a hierarchical structure with labels for Android Development and User Interface Design]
EMS TRAUMA REPORT
EMS TRAUMA REPORT
EMS TRAUMA REPORT
EMS TRAUMA REPORT
EMS TRAUMA REPORT
EMS TRAUMA REPORT

Test Protocols version 1.0

<table>
<thead>
<tr>
<th>Id</th>
<th>Fields</th>
<th>Allowed Data and Test</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>First Name</td>
<td>Required field cannot be blank. This field should accept only letters.</td>
</tr>
<tr>
<td>2</td>
<td>Last Name</td>
<td>Blank field validation, should accept the text.</td>
</tr>
<tr>
<td>3</td>
<td>Password</td>
<td>Blank field validation, minimum 6 characters required. This parameter must be filled with proper data.</td>
</tr>
<tr>
<td>4</td>
<td>Re-Type Password</td>
<td>On miss type password an alert should show the user. Otherwise, it should accept the password.</td>
</tr>
<tr>
<td>5</td>
<td>Address</td>
<td>Blank field validation, insert some address to move on.</td>
</tr>
<tr>
<td>6</td>
<td>Telephone</td>
<td>Only digits are allowed, No characters are allowed.</td>
</tr>
<tr>
<td>7</td>
<td>Email</td>
<td>Should only get the valid email, pattern like <a href="mailto:email@example.com">email@example.com</a>, if the email already exists show some alert.</td>
</tr>
<tr>
<td>8</td>
<td>Sign Up</td>
<td>On Sign Up validate fields and process to sign up, if any field is missing, it shouldn’t take user to the next view.</td>
</tr>
<tr>
<td>9</td>
<td>Cancel</td>
<td>On tap cancel it should take user to the back step.</td>
</tr>
</tbody>
</table>

Login

Login screen enables the user to enter their valid information to interact with the application.

User needs to provide the following information in order to use the application, Field follows:

1. Username (A unique name created at the time of sign up)
2. Password (Valid password created at the time of sign up)
3. Remember Me Checkbox (If checked, it stores the user’s login information and allows the user to log in again when the application is re-launched.)
EMS TRAUMA REPORT

TOP-DOWN RECONFIGURATION

Test Protocols version 1.0

<table>
<thead>
<tr>
<th>Id</th>
<th>Fields</th>
<th>Allowed Data and Test</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>First Name</td>
<td>Required field cannot be blank. This field should accept up to 255 characters.</td>
</tr>
<tr>
<td>2</td>
<td>Last Name</td>
<td>Blank field, should accept the text</td>
</tr>
<tr>
<td>3</td>
<td>Password</td>
<td>Blank field validation, minimum 6 characters required. Type proper data.</td>
</tr>
<tr>
<td>4</td>
<td>Re-Type Password</td>
<td>On miss type password an alert should show the user. Q1 should accept the password.</td>
</tr>
<tr>
<td>5</td>
<td>Address</td>
<td>Blank field validation, insert some address to move one character</td>
</tr>
<tr>
<td>6</td>
<td>Telephone</td>
<td>Only digits are allowed, No characters are allowed.</td>
</tr>
<tr>
<td>7</td>
<td>Email</td>
<td>Should only get the valid email, pattern like <a href="mailto:email@email.com">email@email.com</a> already exists show some alert.</td>
</tr>
<tr>
<td>8</td>
<td>Sign Up</td>
<td>On Sign Up validate fields and process to sign up. If any piece is missing, it shouldn't take user to the next view.</td>
</tr>
<tr>
<td>9</td>
<td>Cancel</td>
<td>On tap cancel it should take user to the back step</td>
</tr>
</tbody>
</table>

Login

Login screen enables the user to enter their valid information to interact with the application. Fields follow:

1. Username (A unique name created at the time of sign up)
2. Password (Valid password created at the time of sign up)
3. Remember Me Checkbox (If checked, it stores the user’s login information and will be remembered the next time the application is re-launched.)
EMS TRAUMA REPORT

TOP-DOWN RECONFIGURATION
EMS TRAUMA REPORT

New High Acuity Case

Basic Information
- Name: John Marc
- Sex: Male
- Time: 07:00 am
- Date: 07/09/15
- Age: 23
- Area: Somewhere

Mechanics of Injury
- Mechanism: Auto vs Bike
- Intubated: Yes
- Respiratory Rate: < 30
- Heart Rate: 80
- Systolic Blood Pressure: 120 / 80

Type of Injury:
- D = Deformity
- C = Contusion
- A = Abrasion
- P = Puncture
- B = Burn
- T = Tendon
- L = Laceration
- S = Swelling
- TQ = Tourniquet
- I = Impaled
- Q = Open Fracture
- G = Gunshot Wound
// = Amputation
How do I become HIPAA compliant? (a check)

By Jason Wang / Published on October 30, 2013

A little housekeeping before we answer the question. This article is not a definitive list of what is required, instead, it should assign a Privacy Officer to review each rule in its entirety. This article is intended to point you in the right direction.

So you have determined that you are handling protected health information (PHI) and that you need to determine next? What steps need to be taken in order to become HIPAA compliant?

1) Written regulation to accept fully identified persons only first and last name
   m) ELABORATION: All users manually approved by an admin before
   or do anything

2) Authentication
   a) RESPONSE: We have an auth system, API requests are authorized
   HTTP Basic (which will be over HTTPS in production), and then
   levels admin (rw on everything) doctor (rw on emergency cases)
   own emergency cases while active, then ro
   b)

3) Logging
   a) RESPONSE: This is something we need to add. I'll be using a
   framework (probably https://github.com/brentv/node-bunyan) to
   log to stdout, and it can be piped into files or whatever in production

HIPAA for trauma app
How do I become HIPAA compliant? (a checklist)

By Jason Wang / Published on October 30, 2013

A little housekeeping before we answer the question. This article is not a definitive list of what is needed and should assign a Privacy Officer to review each rule in its entirety. This article is intended to point you in the right direction.

So you have determined that you are handling protected health information (PHI) and that you need to be HIPAA compliant next? What steps need to be taken in order to become HIPAA compliant?

1) Written regulation to accept fully identified persons only first and last names
   a) ELABORATION: All users manually approved by an admin before or do anything

2) Authentication
   a) RESPONSE: We have an auth system, API requests are authenticated HTTP Basic (which will be over HTTPS in production), and then levels admin (rw on everything) doctor (rw on emergency cases) own emergency cases while active, then no
   b) 

3) Logging
   a) RESPONSE: This is something we need to add. I'll be using a logging framework (probably https://github.com/brentnh/node-bunyan) to log to stdout, and it can be piped into files or whatever in production

HIPAA for trauma app
EMS TRAUMA REPORT
EMS TRAUMA REPORT

Bottom-Up Reconfiguration
EMS TRAUMA REPORT
TRUE STORY GAME

CRUSHING
Subtle looks, pounding pulse
However long the hover lasts
Between friend zone and fun zone
Android companion app spun up in the final week
ENTERPRISE WORKSHOP PORTAL
## ROLE HIRES IN <14 MINUTES

<table>
<thead>
<tr>
<th></th>
<th>EMS Report</th>
<th>True Story</th>
<th>Enterprise Portal</th>
<th>All Projects</th>
</tr>
</thead>
<tbody>
<tr>
<td>Median time</td>
<td>13:40</td>
<td>12:40</td>
<td>15:13</td>
<td>13:40</td>
</tr>
<tr>
<td>(mm:ss)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

20 manual hires in a median 889 minutes (~15 hours)
## RECONFIGURED ORG STRUCTURES

<table>
<thead>
<tr>
<th></th>
<th>EMS Report</th>
<th>True Story</th>
<th>Enterprise Portal</th>
<th>All Projects</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong># of pull requests</strong></td>
<td>335</td>
<td>113</td>
<td>118</td>
<td>566</td>
</tr>
<tr>
<td><strong>Mean pull requests per day</strong></td>
<td>7.3</td>
<td>2.8</td>
<td>3.4</td>
<td>4.5</td>
</tr>
</tbody>
</table>
While organizations could conceivably hire anew for each role, in practice they also accreted members and rotated those members into new roles.
REFLECTIONS

When computation is a mediating layer for work, we can design it to help guide and support peoples’ goals.

Current and future contributions:

Use data, theory, and experimentation to encourage more effective organizational practices.

Change the transaction costs core to the Theory of the Firm.

Extract crowdsourcing from the microtasking swamp.
1) Enable crowd collectives to achieve complex and open-ended goals

2) Recruit effective collaborators despite unpredictable availability

3) Crowdsourcce research itself, providing global access to upward mobility
Huddler

FLASH ORGANIZATIONS SACRIFICE
FAMILIARITY TO GAIN SPEED

Revisiting asset specificity...

Team-based coordination requires that team members become familiar with each other by working together over time [Huckman, Staats, and Upton 2009; Reagans, Argote, and Brooks 2005]
FAMILIAR TEAMS PERFORM BETTER

Teams from AMT authored creative ads for Kickstarter projects

Manipulation: Team membership was random in each round, or kept familiar by maintaining the team across rounds

Measure: AdWords CTR

By Task 5, familiar teams had twice the CTR of random teams: $t(31)=3.37, p<.01, d=1.2$
CROWDSOURCING IS AT ODDS WITH FAMILIARITY

On-demand crowdsourcing would seem to make building familiarity infeasible

**Goal:** a system that enables assembly of familiar crowd teams, even under unpredictable availability and strict time constraints
Create advertisements for Kickstarter projects

Started: 1 hour ago

@megaturker  @lilmisssunshine  @teabreak
Given a time constraint $t$ and a current set of team members, find a schedule of people $p_1...n$ to invite and wait times $t_1...n$ to wait for each person to respond.

Maximize the expected familiarity of the resulting team, given invitees’ probability of being available.

$$\text{maximize} \sum_{i=1}^{n} \left[ \text{availability}(p_i, t_i) \cdot \sum_{p_m \in \text{team}} \text{familiarity}(p_i, p_m) \right]$$

subject to

$$\sum_{i=1}^{n} t_i \leq t$$
Planning who to ask and how long to wait before moving on is a combinatorial problem with an exponential number of alternatives.

Dynamic program: recursively compute sub-solutions

\[
\sum_{i=1}^{n} \left[ \text{availability}(p_i, t_i) \cdot \sum_{p_m \in \text{team}} \text{familiarity}(p_i, p_m) \right]
\]

\[
= E([p_1, \ldots, p_n], t)
\]

\[
= \max_{0 \leq t_1 \leq t} E([p_2, \ldots, p_n], t - t_1) + E(p_1, t_1)
\]
EVALUATION

N=280 teams assembled from Amazon Mechanical Turk over two weeks to perform Kickstarter advertisement design tasks, randomized across condition.

Control | Availability only
Familiarity only | Huddler

Measured: pairwise tasks between team members, minutes to form team.
Huddler convened highly familiar teams nearly as quickly as when only trying to optimize for speed

Two-way ANOVA: significant main effects (p<.01), no significant interaction

Two-way ANOVA: significant main effects (p<.01), significant interaction (p<.01)
REFLECTIONS

Crowdsourcing does not need to give up the social fabric of teamwork in order to achieve rapid, responsive efforts

Current and future contributions:

- Adapting Huddler’s value function to span other goals: personality balancing, diverse expertise, predicted performance, satisfaction...

- Countering Huddler’s risk aversion from availability — the system exploits early teams rather than build a deeper network
1) Enable crowd collectives to achieve complex and open-ended goals

2) Recruit effective collaborators despite unpredictable availability

3) Crowdsource research itself, providing global access to upward mobility
Crowd research

Vaish, Gaikwad, Veit, Krishna, Ibarra, Simoiu, Wilber, Belongie, Davis, Goel, Bernstein.
Ongoing.
RESEARCH: THE DOMAIN OF THE PRIVILEGED FEW

Those able to attend prestigious universities can access research experiences that support open-ended inquiry and launch careers

...but the vast majority of people cannot

A research ecosystem that under-represents minorities and developing regions, and a literature that overlooks their perspectives

Top 50 global universities, US News 2017
CROWD RESEARCH

A crowdsourcing technique enabling a global crowd to work together on an open-ended research project

Participants collaborate as one large team to brainstorm, execute and publish the project under the leadership of a PI
GOALS

Give access to training and research experiences that can enable upward career and educational mobility

Convene hundreds or thousands of people on a single ambitious project
WE ARE NOT EQUIPPED FOR LARGE-SCALE OPEN-ENDED RESEARCH

Research is not a linear path from idea to result: it is an iterative process of exploration

[Gowers 2000]

In contrast, citizen science efforts today focus on pre-defined goals in order to structure the crowd’s contributions

[Cranshaw and Kittur 2011]

[Cooper et al. 2010]
PROBLEMS

**Coordination:**
How do we prevent the project from moving in 1,000 directions at once, across easily 6,000 messages per week?

**Credit:**
How can we provide proof that participants made substantial contributions to the project, when no one central authority can assert this?
Iterative crowdsourcing technique:
Weekly cycle of open contribution, synchronous collaboration, and peer assessment

Decentralized credit:
Participants allocate finite credits to each other, enabling a graph centrality algorithm to determine credit and author order
CROWDSOURCING PROCESS

open call → group meeting → milestone deadline → peer assessment
12-85 weeks
500,000 Slack messages
190,000 minutes of video meetings
Task Planning

Milestones

Getting Started

Understanding lives of workers

Try being a worker on oDesk (now Upwork.com) or other large project platforms.

Hello, world! Getting started with our code

Work on one of our open feature requests on GitHub

Get your hands dirty and set up our Hello, World example

Related work/papers: read and comment

Read the MobileWorks paper

Read Flash Teams paper

Read paper on the future of...
ENGINEERING

MILESTONES

Apr 5, 2015 – Oct 26, 2015
Contributions to develop2, excluding merge commits

Contributions: Commits

<table>
<thead>
<tr>
<th>Author</th>
<th>Commits</th>
<th>Added</th>
<th>Removed</th>
</tr>
</thead>
<tbody>
<tr>
<td>dmorina</td>
<td>502</td>
<td>141,296</td>
<td>365,266</td>
</tr>
<tr>
<td>nistala</td>
<td>249</td>
<td>1,671,624</td>
<td>1,442,878</td>
</tr>
</tbody>
</table>

#1

#2
PROTOTYPING

MILESTONES

A worker has a question about a posted task, and emails the requester.

Requester answers to ensure work is done as required.

Both parties are satisfied.

The profiles are utilized to clear up confusion before further resolution is required.

Both parties are satisfied.
DATA ANALYSIS

MILESTONES
1. Anyone can pitch an idea. If it gets enough support, it goes to the next election and needs majority support from both workers and requesters.
   - *(original)* **Direct democracy:** anyone can pitch a policy idea, and once it gets past a threshold of support (e.g., 1000 votes), it goes up on a ballot. Twice a year, ideas go out to a direct vote for everyone on the platform. If it gets majority support from both workers and requesters, it passes.
2. Members get elected as worker or requester representatives (3 each) to a panel. Tiebreaking from a 7th member (jointly elected president).
   - *(original)* **Representative democracy:** once a year, members of the platform can be elected as either worker or requester representatives for a small panel (e.g., six people). Anybody can pitch a policy idea, and once it gets past a threshold of support (e.g., 1000 votes), the elected representatives must discuss it and vote on it.
3. *Wikimocracy:* the site's rules and policies are a wiki. Anyone can discuss, and if they edit, policies change directly.
4. Any idea that gets enough support enters a public one-month voting period. It's completely voluntary to vote. (Like a Kickstarter campaign.)
   - **Original:** **Fast-paced referendums:** similar concept as direct democracy, but instead of per year, you do it as vote thresholds within a month (within time of posting), and it's completely voluntary to vote. Kinda like a campaign on Kickstarter. Fast pace and flexible deadlines will help the ideas continually flowing in.
5. For low-level changes, highlight the interface and suggest changes directly.
   - Upvote/downvote directly on the interface.
PEER ASSESSMENT

MILESTONES

- **Sustainable Reputation Mechanism** by Neil
- **Chat and Rating Prototypes for Crowdsourcing Social Media Platform** by Kristine Hoang
- **Leveling, Rating and Categorisation** by Soroosh Bateni
- **Worker Requester Mentorship** by Mike Young
- **EmpathySociety: Triggering Empathy via Smart Mechanisms!** by Saiph Savage
- **Compensation Suggestion Tool** by Trygve Cossette
Andrew Ng, Stanford and Baidu Research
DECENTRALIZED CREDIT: TURN IT INTO A GRAPH PROBLEM

Each participant allocates 100 credit points to other participants based on their assessment of who impacted the project.

Resulted: weighted directed graph
GRAPH CENTRALITY: PAGERANK

Intuition: identify nodes that are receiving large amounts of credit, weigh those nodes’ allocations heavily, and iterate until convergence

Propagate each node’s score in proportion to its outgoing wedge weights
STRATEGIC BEHAVIOR

Speaking different languages or otherwise interacting with only a small part of the crowd: link ring

Strategically directing credit toward those who will return credit to you: such attacks occur in 360-degree reviews

Formulations of centrality algorithms such as PageRank can correct for most of these attacks
1500 participants from six continents

2% high school, 73% undergrad, 22% master’s, 3% PhD
RECRUITMENT: PROVIDING ACCESS

Matching names to DBLP: 90% with no prior research experience

Matching affiliations to Times Higher Education Global Rankings: 75% come from universities ranked below 500

Participants have gone on to programs at Stanford, UC Berkeley, and Carnegie Mellon University, and MIT
LARGE-SCALE PROJECTS

Design and develop a new paid crowdsourcing platform

Michael Bernstein, Stanford, HCI

Run hundreds of parallel experiments

Sharad Goel, Stanford, Data Science

Create new hybrid human-computer vision algorithms

James Davis, UCSC, and Serge Belongie, Cornell Tech, Computer Vision
Boomerang: Rebounding the Consequences of Reputation Feedback on Crowdsourcing Platforms


Stanford Crowd Research Collective, Stanford University
daemo@cs.stanford.edu

ABSTRACT
Paid crowdsourcing platforms suffer from low-quality work and unfair rejections, but paradoxically, most workers and requesters have high reputation scores. These inflated scores, which make high-quality work and workers difficult to find, stem from social pressure to avoid giving negative feedback. We introduce Boomerang, a reputation system for crowdsourcing platforms that elicits more accurate feedback by rebounding the consequences of feedback directly back onto the person who gave it. With Boomerang, requesters find that their highly-rated workers gain earliest access to their future tasks, and workers find tasks from their highly-rated requesters at the top of their task feed. Field experiments verify that Boomerang causes both workers and requesters to provide feedback that is more closely aligned with their private opinions. Inspired by a game-theoretic notion of incentive-compatibility, Boomerang opens opportunities for interaction design to incentivize honest reporting over strategic dishonesty.

INTRODUCTION
Today's crowdsourcing platforms are markets for lemons. On crowdsourcing platforms such as Amazon Mechanical Turk and Upwork, both workers and requesters face significant uncertainty: workers struggle to predict whether a requester will pay for their work, and requesters worry whether workers will produce high-quality results. To reduce this uncertainty, crowdsourcing platforms rely on reputation systems such as task acceptance rates for workers and star ratings for requesters. However, reputation systems are unable to address this uncertainty: for example, workers' acceptance rates on Mechanical Turk are often above 97%, regardless of the quality of their work. The result is a downward spiral: requesters offer lower wages to offset their risk of low-quality results, and workers respond to lower payment with lower quality work. Ultimately, ineffective reputation systems lead to dissatisfaction, lost productivity, and abandonment.

Reliable reputation information does exist—workers and requesters form detailed opinions about each other as they interact—but this information is difficult to elicit accurately. The core challenge is reputation inflation: significant social costs to providing negative feedback cause people to "generously round up" their feedback scores, even if they hope to never work with that partner again. Incentives are misaligned: for example, by the time a worker realizes that a requester is low-quality, the worker is already mid-way through the task and has little reason to report the issue accurately.

This text field is large enough to hold the appropriate release statement assuming it is single spaced.

Every submission will be assigned their own unique DOI string to be included here.
Crowd Guilds: Worker-led Reputation and Feedback on Crowdsourcing Platforms

Mark E. Whiting, Dilrukshi Gamage, Snehalkumar (Neil) S. Gaikwad, Aaron Gilbee, Shirish Goyal, Alipta Ballav, Dinesh Majeti, Nalin Chhibber, Angela Richmond-Fuller, Freddie Vargus, Tejas Seshadri Sarma, Varshine Chandrakanthan, Teogenes Moura, Mohamed Hashim Salih, Gabriel Bayomi Tinoco Kalejaiye, Adam Ginzberg, Catherine A. Mullings, Yoni Dayan, Kristy Milland, Henrique Orefice, Jeff Regino, Sayna Parsi, Kunz Mainali, Vibhor Sehgal, Sekandar Matin, Akshansh Sinha, Rajan Vaish, Michael S. Bernstein
Stanford Crowd Research Collective
daemo@cs.stanford.edu

ABSTRACT
Crowd workers are distributed and decentralized. While decentralization is designed to utilize independent judgment to promote high-quality results, it paradoxically undercuts behaviors and institutions that are critical to high-quality work. Reputation is one central example: crowdsourcing systems depend on reputation scores from decentralized workers and requesters, but these scores are notoriously inflated and uninformative. In this paper, we draw inspiration from historical worker guilds (e.g., in the silk trade) to design and implement crowd guilds, a new system of crowdsourced work evaluation where workers review each other anonymously. In a two-week field experiment, we compared crowd guilds to a traditional decentralized crowd work model. Crowd guilds produced reputation signals more strongly correlated with ground-truth worker quality than signals available on current crowd working platforms, and more accurate than in the traditional model.

Author Keywords
crowdsourcing platforms; human computation

ACM Classification Keywords
H.5.3. Group and Organization Interfaces

INTRODUCTION
Crowdsourcing platforms such as Amazon Mechanical Turk decentralize their workforce, designing for distributed, independent work [16, 42]. Decentralization aims to encourage accuracy through independent judgement [59]. However, by making communication and coordination more difficult, decentralization disempowers workers and forces worker collectives off-platform [41, 64, 16]. The result is disenfranchisement [22, 55] and an unfavorable workplace environment [41, 42]. Worse, while decentralization is motivated by a desire for high-quality work, it paradoxically undercuts behaviors and institutions that are critical to high-quality work. In many traditional organizations, for example, centralized worker coordination is a keystone to behaviors that improve work quality, including skill development [2], knowledge management [35], and performance ratings [58].

In this paper, we focus on reputation as an exemplar challenge that arises from worker decentralization: effective reputation signals are traditionally reliant on centralized mechanisms such as performance reviews [58, 23]. Crowdsourcing platforms rely heavily on their reputation systems, such as task acceptance rates, to help requesters identify high-quality workers [22, 43]. On Mechanical Turk, as on other on-demand platforms such as Upwork and Uber, these reputation scores are derived from decentralized feedback from independent requesters. However, the resulting reputation scores are notoriously inflated and uninformative. In this paper, we draw inspiration from historical worker guilds (e.g., in the silk trade) to design and implement crowd guilds, a new system of crowdsourced work evaluation where workers review each other anonymously. In a two-week field experiment, we compared crowd guilds to a traditional decentralized crowd work model. Crowd guilds produced reputation signals more strongly correlated with ground-truth worker quality than signals available on current crowd working platforms, and more accurate than in the traditional model.
On Optimizing Human-Machine Task Assignments

Abstract
Crowdsourcing systems are at the core of online marketplaces, where workers are engaged to complete varied tasks. However, widely used crowdsourcing platforms have thus far failed to ensure high-quality results, fair wages, and respect for workers. On the other hand, requesters do not trust the results they receive, resulting in an asymmetrical relationship between workers and requesters. Moreover, existing marketplaces suffer from uneven distribution of power, with requesters holding too much influence over workers.

In this work, we present Daemo, a self-governed crowdsourcing marketplace, where workers are responsible for creating and managing marketplaces to complete tasks. Daemo aims to achieve equitable representation of workers and requesters, and to ensure that workers are paid fairly for their work. We envision Daemo as a platform where workers can create, manage, and monitor marketplaces to complete tasks. Daemo includes a system for managing marketplaces, where workers are responsible for creating and managing marketplaces. Daemo also includes a system for monitoring marketplaces, where workers are responsible for ensuring that workers are paid fairly for their work.

Daemo includes a system for managing marketplaces, where workers are responsible for creating and managing marketplaces. Daemo also includes a system for monitoring marketplaces, where workers are responsible for ensuring that workers are paid fairly for their work.

Related Work

Crowdsourcing marketplaces provide opportunities for workers to complete varied tasks. However, existing marketplaces have thus far failed to ensure high-quality results, fair wages, and respect for workers. On the other hand, requesters do not trust the results they receive, resulting in an asymmetrical relationship between workers and requesters. Moreover, existing marketplaces suffer from uneven distribution of power, with requesters holding too much influence over workers.

In this work, we present Daemo, a self-governed crowdsourcing marketplace, where workers are responsible for creating and managing marketplaces to complete tasks. Daemo aims to achieve equitable representation of workers and requesters, and to ensure that workers are paid fairly for their work. We envision Daemo as a platform where workers can create, manage, and monitor marketplaces to complete tasks. Daemo includes a system for managing marketplaces, where workers are responsible for creating and managing marketplaces. Daemo also includes a system for monitoring marketplaces, where workers are responsible for ensuring that workers are paid fairly for their work.

Daemo includes a system for managing marketplaces, where workers are responsible for creating and managing marketplaces. Daemo also includes a system for monitoring marketplaces, where workers are responsible for ensuring that workers are paid fairly for their work.

Acknowledgments

This work was funded by a NSF Career award (IIS-1553769) and a Stanford CMU-HEP fellowship. We thank the anonymous reviewers for their helpful feedback.

Daemo: A Self-Governed Crowdsourcing Marketplace

Alik Shankar Myers‡
PFNstitute of Technology
vikas s yadgar
National Institute of Technology Karnataka

Camelia Simo‡
Stanford University

Sharad Goel‡
Stanford University

Additional Authors

vicky sylajge
National Institute of Technology Karnataka
vicky.sylajge@v.uker.ac.in

Imad Matar‡
Stanford University
imad@stanford.edu

ACM Classification Keywords
Crowdsourcing, open-governance, HCI

INTRODUCTION

Crowdsourcing marketplaces have become ubiquitous in recent years. These marketplaces engage workers to complete varied tasks. However, existing marketplaces have thus far failed to ensure high-quality results, fair wages, and respect for workers. On the other hand, requesters do not trust the results they receive, resulting in an asymmetrical relationship between workers and requesters. Moreover, existing marketplaces suffer from uneven distribution of power, with requesters holding too much influence over workers.

We present Daemo, a self-governed crowdsourcing marketplace, whose workers are responsible for creating and managing marketplaces to complete tasks. Daemo aims to achieve equitable representation of workers and requesters, and to ensure that workers are paid fairly for their work. We envision Daemo as a platform where workers can create, manage, and monitor marketplaces to complete tasks. Daemo includes a system for managing marketplaces, where workers are responsible for creating and managing marketplaces. Daemo also includes a system for monitoring marketplaces, where workers are responsible for ensuring that workers are paid fairly for their work.

We believe that Daemo can serve as a model for future self-governed crowdsourcing marketplaces. Our work is motivated by the need to create a more equitable and transparent system for completing tasks. We hope that Daemo will serve as a platform for workers to create, manage, and monitor marketplaces to complete tasks. We believe that Daemo can serve as a model for future self-governed crowdsourcing marketplaces. Our work is motivated by the need to create a more equitable and transparent system for completing tasks. We hope that Daemo will serve as a platform for workers to create, manage, and monitor marketplaces to complete tasks. We believe that Daemo can serve as a model for future self-governed crowdsourcing marketplaces. Our work is motivated by the need to create a more equitable and transparent system for completing tasks. We hope that Daemo will serve as a platform for workers to create, manage, and monitor marketplaces to complete tasks. We believe that Daemo can serve as a model for future self-governed crowdsourcing marketplaces. Our work is motivated by the need to create a more equitable and transparent system for completing tasks. We hope that Daemo will serve as a platform for workers to create, manage, and monitor marketplaces to complete tasks. We believe that Daemo can serve as a model for future self-governed crowdsourcing marketplaces. Our work is motivated by the need to create a more equitable and transparent system for completing tasks. We hope that Daemo will serve as a platform for workers to create, manage, and monitor marketplaces to complete tasks. We believe that Daemo can serve as a model for future self-governed crowdsourcing marketplaces. Our work is motivated by the need to create a more equitable and transparent system for completing tasks. We hope that Daemo will serve as a platform for workers to create, manage, and monitor marketplaces to complete tasks. We believe that Daemo can serve as a model for future self-governed crowdsourcing marketplaces. Our work is motivated by the need to create a more equitable and transparent system for completing tasks. We hope that Daemo will serve as a platform for workers to create, manage, and monitor marketplaces to complete tasks. We believe that Daemo can serve as a model for future self-governed crowdsourcing marketplaces. Our work is motivated by the need to create a more equitable and transparent system for completing tasks. We hope that Daemo will serve as a platform for workers to create, manage, and monitor marketplaces to complete tasks. We believe that Daemo can serve as a model for future self-governed crowdsourcing marketplaces. Our work is motivated by the need to create a more equitable and transparent system for completing tasks. We hope that Daemo will serve as a platform for workers to create, manage, and monitor marketplaces to complete tasks. We believe that Daemo can serve as a model for future self-governed crowdsourcing marketplaces. Our work is motiva...
THE CROWD LED IDEATION RESULTS

Thematic coding of milestone submissions across weeks
THE CROWD INVESTED TIME

RESULTS
THE CROWD LED WRITING

RESULTS

Count of number of edits to shared document

Crowd: 80%

Principal investigator: 20%
ANALYZING PAGERANK’S EFFECT

What impact did PageRank have on credit distribution?

Method: normalize raw summed credit scores, and PageRank-adjusted scores, to sum to 1.0

Regress both raw score and PageRank score on observable collaboration behaviors, and compare $\beta$ estimates across the regressions.
## LESS TALKING, MORE DOING

<table>
<thead>
<tr>
<th>Participation Measure</th>
<th>PageRank: $\beta_{\text{PR}}$</th>
<th>Raw Votes: $\beta_{\text{raw}}$</th>
<th>$\beta_{\text{PR}} - \beta_{\text{raw}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td># Hangouts</td>
<td>0.0694***</td>
<td>0.0438*</td>
<td>0.0256</td>
</tr>
<tr>
<td># Files Uploaded</td>
<td>0.0352**</td>
<td>0.0293*</td>
<td>0.0059</td>
</tr>
<tr>
<td># GitHub commits</td>
<td>0.0171</td>
<td>-0.024*</td>
<td>0.0411***</td>
</tr>
<tr>
<td># Slack messages</td>
<td>0.0351*</td>
<td>0.1122***</td>
<td>-0.0770***</td>
</tr>
<tr>
<td># self-organized meetings</td>
<td>0.0239*</td>
<td>0.0115</td>
<td>0.0123</td>
</tr>
<tr>
<td>Milestone leader (binary)</td>
<td>0.0360***</td>
<td>0.0059</td>
<td>0.0300**</td>
</tr>
<tr>
<td>Weeks active</td>
<td>0.0252*</td>
<td>0.0141</td>
<td>0.011</td>
</tr>
</tbody>
</table>

All variables standardized
EFFECTS ON AUTHOR ORDER

Pagerank

Raw vote ranking

#2 and #4 have a high rank due to link ring
EFFECTS ON AUTHOR ORDER

PAGERANK

PageRank-corrected author order
Influential coauthors reduced impact of link ring
REFLECTIONS

Computation and crowdsourcing can scale not just the teaching of new skills and the execution of research, but the experience of research and upward career mobility as well.

Current and future contributions:

- Decentralized evaluation could help even traditional groups escape the tyranny of top-down review.
- Projects that not only reach more people, but operate at a larger technical scale than traditional CS research.
Rather than structuring crowds like algorithms, let’s structure them like organizations.

Organizations were originally designed with inspiration by mechanical systems. What might a computational infrastructure offer them?
In A Flash:
Crowdsourcing Organizations, Collaboration, and Research

Thanks to...
Amazing students: Daniela Retelny, Niloufar Salehi, Rajan Vaish
Amazing colleagues: Melissa Valentine, Sharad Goel, James Davis
Amazing supporters: NSF, Sloan, Accenture, Microsoft, Toyota, Stanford Cyber Initiative, ONR, HPDTRP
Amazing questions