Crowd-Powered Interfaces

Michael S. Bernstein
MIT CSAIL
Cambridge, MA
msbernst@csail.mit.edu

ABSTRACT
We investigate *crowd-powered interfaces*: interfaces that embed human activity to support high-level conceptual activities such as writing, editing and question-answering. For example, a crowd-powered interface using paid crowd workers can compute a series of textual cuts and edits to a paragraph, then provide the user with an interface to condense his or her writing. We map out the design space of interfaces that depend on outsourced, friendsourced, and data mined resources, and report on designs for each of these. We discuss technical and motivational challenges inherent in human-powered interfaces.

ACM Classification: H5.2 [Information interfaces and presentation]: User Interfaces. - Graphical user interfaces.

General terms: Design, Human Factors

Keywords: Social computing, outsourcing, crowdsourcing

INTRODUCTION

There is a gulf between the complex, high-level tasks that interfaces try to support and the simple levers they can actually provide. Complex cognition tasks like writing a paper are a good example: word processors can help with layout, but can't help with writing decisions because they don't deeply understand the text. They also cannot automate complicated actions like changing the paper's tone to be more formal. The interface only helps chain together low-level actions like formatting or correction. Current interfaces act more like a toolbox than a personal assistant.

The information and computation is available to power these high-level interfaces — it is embedded in humans. Other authors can provide writing and editing support for a paper; expert writers' high-level patterns can inform novices' documents; a writer's friends would know whether colleagues' names had been spelled correctly. If we could use this information or reproduce the humans' computation, we could create interfaces that support user tasks at a much higher level. Of course, this is eminently difficult to do: the people with the knowledge have limited time to help others.

This thesis contributes interfaces that utilize human contributions to support complex cognitive tasks and actions. They are *crowd-powered interfaces*: interfaces that rely on human activity traces or human computation [8] to provide benefit to the end user. Crowd-powered interfaces motivate or pay human workers to complete tasks that power complex interactions. For example, paid crowd workers might suggest edits to tighten up writing, providing the user with a mechanism to drag a slider to shorten the article to the desired length. If past human activity traces are available, they can embed that activity to lower cost and latency.

In support of crowd-powered interfaces, we propose new algorithms and social incentives for crowd control. Lazy crowd workers will do as little work as possible, while overeager ones will make unexpected (or worse, undesired) changes. We introduce design patterns for controlling these problems. For example, we decompose the work into separate stages for finding problems, fixing problems, and performing quality control. Motivation is also a difficult problem, so we utilize social connectedness as a motivator.

UIST'10, October 3–6, 2010, New York, New York, USA.

DESIGN SPACE

The design space of crowd-powered interfaces is vast. In this work we explore this design space through a series of prototype systems, set around themes of data collection.

1. *Friendsourcing:* Crowd-powered interfaces can pursue specialized applications by finding users with domain knowledge. In particular, we introduce the means to incentivize members of a user's social network to share information about the user to personalize that user's experience.
2. *Outsourcing:* Paid crowd workers can power complex interfaces that require generic human cognition. We have built a word processor plug-in where crowd workers power interactions like shortening long sections of text. Because interaction often depends on immediate reactions to the user's input, we plan to explore crowd-powered interfaces that place a crowd worker in a real-time support position for the user.
3. *Data mining:* Not all crowd-powered interfaces require users to perform new work. Instead, we can remix activity traces from other users. We investigate ways that work from expert users might enable novices to perform nearer expert levels.

The remainder of this paper fleshes out the visions of outsourced, friendsourced and crowd-powered interfaces.

FRIENDSOURCING: PERSONALIZED INTERFACES

Social data is critical to many interactive applications. For example, Yahoo! Answers cannot easily help with questions about the history of your small a cappella group or the

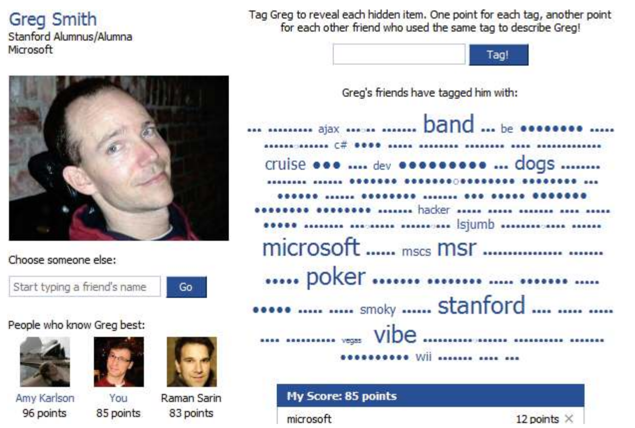

Figure 1. In Collabio, this user has guessed several tags for Greg Smith, including *band, ohio* and *vegas*.

way your friend delivered his marriage proposal. When information is known only to a small group in a social network, traditional crowdsourcing mechanisms struggle to motivate a large enough user population to share.

In such situations, we bring social application design to bear via an approach we call *friendsourcing* [3]. Friendsourcing gathers social information in a social context, incentivizing a user's social network to share information or perform work. Here, we friendsource for personalization, gathering descriptive information about a person that we can use to power interactive applications on their behalf. We have designed systems to collect friendsourced data, as well as a series of interactive prototypes using the data.

Collabio

We developed Collabio (*Collab*orative *Bio*graphy), a game that elicits descriptive tags for individuals within Facebook [2, 3]. Collabio (Figure 1) elicits information that your friends know about you. This information includes your personality, expertise, artistic and musical tastes, topics of importance, and quirky habits. The application leverages properties of the social network such as competition and social accountability to solve the tag motivation and accuracy problems within a social framework.

The main activity of Collabio is tagging friends. Players see a tag cloud that other friends have collectively authored by tagging the selected friend. When presenting this cloud, Collabio hides tags that the user has not guessed. Each hidden tag has its letters replaced with solid circles; for example, the tag *UIST* appears as ●●●●. When a user guesses a tag that others also guessed, it is revealed within the cloud.

Figure 3. Collabio Clouds comparing users tagged with *washington* to users tagged with *georgia tech*.

Figure 2. The FeedMe plug-in for Google Reader suggests friends, family, and colleagues who might be interested in seeing the post that you are reading.

For each guess, users receive points equal to the number of people who have applied a tag, including themselves. If they are the only person who guessed that tag, then they get 1 point; if there are 11 others, they get 12 points.

Collabio has gathered 7,780 unique tags on 3,831 individuals in 29,307 tagging events during its deployment.

FeedMe

Collabio encourages users to take on a new behavior – friend tagging – but we can do better by augmenting existing behaviors. Today, to find interesting web content, people rely on friends and colleagues to pass links along as they encounter them. In the FeedMe project [1], we study and augment link-sharing via e-mail, the most popular means of sharing web content today. We have developed FeedMe (Figure 2), a plug-in for Google Reader that makes directed sharing of content a more salient part of the user experience. FeedMe recommends friends who may be interested in seeing content that the user is viewing, provides information on what the recipient has seen and how many emails they have received recently, and gives recipients the opportunity to provide lightweight feedback when they appreciate shared content. It also introduces a novel design space within mixed-initiative social recommenders: friends who know the user vet the material on the user's behalf.

By making sharing easier, FeedMe can implicitly learn user models by tracking which articles are sent to which friend. It utilizes a user's social network to produce accurate user models without the user's involvement.

Applications of Friendsourcing

Friendsourced data can open up new avenues for interactive applications. We have produced a pair of novel crowd-powered interfaces using the Collabio tag database: a tag cloud aggregator for tag visualization and exploration and an expert-finding question answering system.

Figure 4. Collabio QnA is a question and answer system that uses Collabio tags to find friends and friends-of-friends who can answer your questions.

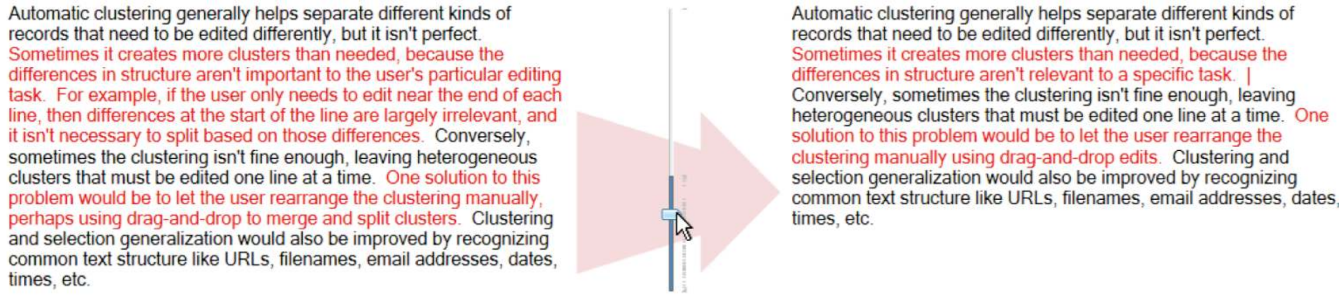

Figure 5. Shortn allows users to adjust the length of a paragraph via a slider. Red text indicates locations where cuts or rewrites have occurred. Tick marks represent possible lengths, and the blue background bounds the possible lengths.

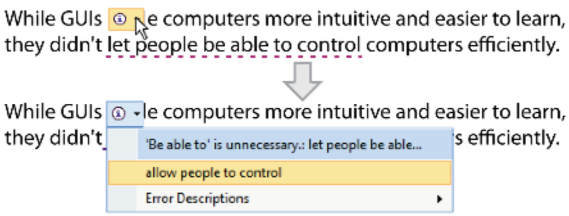

Figure 6. Crowdproof is a human-augmented proofreader. The drop-down explains the problem (blue title) and suggests fixes (gold selection).

Collabio Clouds (Figure 3) aggregates tag clouds based on user queries [3]. The user can query his or her own tag cloud as well as the aggregated tag cloud of friends, Collabio users, users tagged with specific Collabio tags (like *tennis* or *Adobe*), or users in Facebook networks or groups. Collabio Clouds allows users to explore questions such as: What do the tag clouds of members of the Penn State network look like? What other tags show up on individuals tagged with *machine learning*?

The second application is an expert-finding system [3]. Question and answer (QnA) systems such as Yahoo! Answers rely on a large community of answerers actively seeking out questions. Collabio tags and the social network context provide the opportunity for our QnA system to route questions more highly relevant within the user's social network, such as *When is the next HCI group meeting?*, or *Who might be interested in starting an IM football team at Google?* Users ask questions, and Collabio QnA (Figure 4) searches over the Collabio tags to identify friends and friends-of-friends who are most likely to be able to answer the question, then asks the requested individuals.

OUTSOURCING: BATCHED AND REALTIME

When an interface can rely on generic human cognition and friends cannot be motivated to help, paid crowd workers can step in. In this section, we describe an interface that embeds Amazon Mechanical Turk (www.mturk.com) workers to support word processing tasks, and outline a future realtime system.

Soylent

We have developed Soylent, a word processing interface that utilizes crowd contributions to aid complex writing tasks [4]. *Soylent is people*: its core algorithms involve calls to Mechanical Turk workers (Turkers). The system is comprised of three main components.

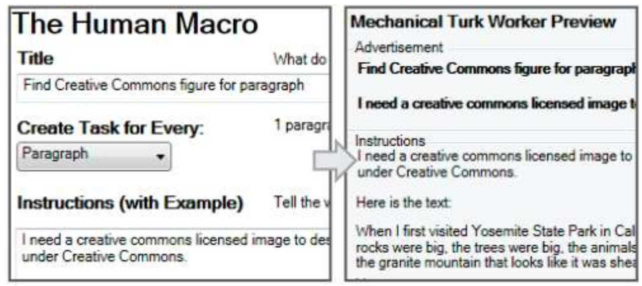

Figure 7. The Human Macro is an end-user programming interface for automating document manipulations. The left half is the user's authoring interface; the right half is a preview of what the Turker will see.

Shortn (Figure 5) is a text shortening service that typically cuts selected text down to 85% of its original length without changing the meaning of the text or introducing errors. The user selects the area of text that is too long, then presses the Shortn button to launch Mechanical Turk tasks in the background that overly wordy parts of the text and propose rewrites. Launching the Shortn dialog box (Figure 5), the user sees the original paragraph on the left. On the right, Shortn provides a single slider to continuously adjust the length of the final paragraph. As the user drags the slider, Shortn computes the combination of crowd trimmings that most closely match the desired length. From the user's point of view, as she moves the slider to make the paragraph shorter, sentences are slightly edited, combined and cut to match the desired length.

Second, Crowdproof (Figure 6) is a human-powered spelling and grammar checker that finds problems Word misses, explains the problems, and suggests fixes. Mechanical Turk workers read sections of text to find and fix errors that Word does not correct. It calls out edited sections with a purple dashed underline. If the user clicks on the error, a drop-down menu explains the problem and offers a list of alternatives.

The Human Macro is an interface for offloading arbitrary word processing tasks. For example, we have used The Human Macro to format citations correctly and find figures for paragraphs. Launching the Human Macro opens a request form (Figure 7). The form dialog is split in two mirrored pieces: a task entry form on the left, and a preview of what the Turker will see on the right. The user then chooses how many separate Turkers he would like to complete the task, and whether the Turkers' work should replace the existing text or annotate it with comments.

Proposal: Realtime Support

Soylent is powerful but suffers from the limitation that its interactions are only enabled after a significant time delay. Our next step is a prototype system with real-time crowd-powered interaction. This system will pursue at least one of two mechanisms: 1) The Surgeon's Assistant, placing a paid crowd worker in a position to aid the user directly, for example by manipulating or customizing the interface as the user focuses on the task; 2) The Quantum User, where crowd workers explore future possible states of the user interface and their actions are used to "backfill" the user's task when the user arrives on a path that a Turker explored.

Technical Challenges

Mechanical Turk costs money and it can be error-prone; to be worthwhile to the user, we must control costs and ensure correctness. We have developed a crowd programming design pattern called Find-Fix-Verify that splits complex crowd intelligence tasks into a series of generation and review stages. These stages use independent agreement and voting to produce reliable results. Rather than ask a single crowd worker to read and edit an entire paragraph, for example, Find-Fix-Verify recruits one set of workers to find candidate areas for improvement, then collects a set of candidate improvements, and finally filters out incorrect candidates. This process prevents errant crowd workers from contributing too much, too little, or introducing errors.

With respect to realtime crowd support, we expect to pursue technical means to lower latency and cost, for example multiplexing each Turker across several users at once.

DATA MINING: REMIXING EXPERTS

We know that we can incentivize a social group to share information that will power an interface, and that (by using payment as an incentive) we can embed live humans in interfaces to produce new interactive possibilities. However, these interfaces require the crowd to do up-front work, and that work is only useful to one user.

In the final piece of the thesis, we propose interaction possibilities that arise when we aggregate multiple user traces. In particular, usage patterns of expert users can be leveraged to aid the average user. The final result of this interface will be a system that allows users to seamlessly stitch together expert-created data to create output that they would not otherwise have been able to create. This work builds on previous research which aggregates activity traces or datasets to produce novel interfaces (e.g., [5, 6, 7]).

Critically, this research still needs to identify a task to support. One hypothetical system is a smart camera trained on top photos from Flickr. As the user composes a shot, the camera compares the photo to a database of quality photos in terms of light, color, orientation, subject placement, and other machine-viewable features. The camera can then recommend actions like zooming in, moving the photograph subject, or adjusting the room lighting. The user can interactively adjust the shot as the camera provides feedback.

ONGOING WORK

Of the systems described in this paper, FeedMe, Collabio, and Soylent have been published. The remaining two – the realtime outsourced support system (The Surgeon's Assistant) and the expert aggregation interface – are still in early stages of design. We hope that participation in the doctoral colloquium will help us refine the map of uncharted space this thesis purports to explore and will provide feedback on the eventual designs of these prototype systems.

CONTRIBUTIONS

This thesis will demonstrate that social data from a crowd, a social network site, and a labor market can each be used to produce user interfaces that are more intuitive and powerful than the ones we use today. We believe that crowd-powered interfaces can support high-level tasks like personalization and writing by substituting human cognition where algorithms and interfaces cannot succeed yet. We make contributions in three areas: 1) we introduce novel applications areas for *interactive systems design*; 2) we develop novel *human computation algorithms* to manage contributors; 3) we *design social systems* that incentivize group members to contribute desired information.

CONCLUSION

We pursue the design of crowd-powered interfaces: interfaces that integrate human activity and cognition to support complex end-user tasks. Data and algorithms for these interfaces may be outsourced to a labor market like Mechanical Turk, friendsourced to the user's social network, or crowd-powered to a large group of Internet users interacting with the same software. We demonstrate prototype systems for each of these applications areas that support needs which AIs and user interface design have not yet accomplished.

REFERENCES

1. Bernstein, M., Marcus, A., Karger, D.R., and Miller, R.C. Enhancing Directed Content Sharing on the Web. *CHI '10*, ACM Press (2010).
2. Bernstein, M., Tan, D., Smith, G., et al. Collabio: A Game for Annotating People within Social Networks. *UIST '09*, ACM Press (2009), 177–180.
3. Bernstein, M., Tan, D., Smith, G., et al. Personalization via Friendsourcing. *ACM Transactions on Human-Computer Interaction (TOCHI) 17*, 2 (2010), 1-28.
4. Bernstein, M.S., Little, G., Miller, R.C., et al. Soylent: A Word Processor with a Crowd Inside. *UIST '10*, ACM Press (2010).
5. Chen, T., Cheng, M., Tan, P., et al. Sketch2Photo. *ACM Transactions on Graphics 28*, 5 (2009), 1.
6. Hartmann, B., MacDougall, D., Brandt, J., and Klemmer, S. What Would Other Programmers Do? Suggesting Solutions to Error Messages. *CHI '10*, ACM Press (2010).
7. Simon, I., Morris, D., and Basu, S. MySong: automatic accompaniment generation for vocal melodies. *Proc. CHI '08*, ACM Press (2008).
8. von Ahn, L. Human computation. *International Conference On Knowledge Capture*, (2007).