April 3, 2011

The Crowdsourcing Compiler

Michael Kearns
Computer and Information Science
University of Pennsylvania
mkearns@cis.upenn.edu

The past several years have provided a growing number of successful examples of the phenomenon broadly known as "crowdsourcing" or "peer production": the deliberate (or sometimes even inadvertent) marshalling of large numbers of online volunteers in service of the creation of a powerful technology-based service or artifact. Among the more visible instances are the Web itself, Wikipedia, the ESP game, deli.cio.us, social networking services, online labor markets such as Amazon's Mechanical Turk, the DARPA Balloon Challenge, and FoldIt, a protein folding game. Every week witnesses both new startups betting almost entirely on crowdsourced contributions, or online grass-roots movements that simply build something powerful quickly, and from the bottom up.

While the list of such apparently successful examples grows, we actually have very poor --- in fact, almost no --- understanding of the phenomenon of crowdsourcing from a design and engineering standpoint: all we can do is look at what has already been done, and guess what might work in a new setting. But this is not how we design computers or systems, where we have well-understood and tested design principles, successful abstractions, notions of modularity that allow componentwise design and construction, and so on. We thus propose that a similar design and engineering discipline is both interesting and necessary for the advancement of crowdsourcing. We posit that massive volunteer forces (both online and in the physical world) can be viewed as a valuable computing substrate for crowdsourced systems and services, and we should begin to create a science for how to best use such a powerful resource for specific tasks. Central to this goal will be two fundamental issues: understanding the *incentives* under which volunteers will work most effectively; and understanding how best to *organize* volunteers in service of a particular task. The research will necessarily be interdisciplinary, requiring ideas and methods from computer science, game theory and economics (both traditional and behavioral), sociology, and many other fields.

A Grand Challenge for this endeavor might be called the *Crowdsourcing Compiler:* the development of high-level programming languages for specifying large-scale, distributed tasks whose solution requires combining traditional computational and networking resources with volunteer human intelligence and contributions. The compiler would translate an abstract program into a more detailed organizational plan for machines and people to jointly carry out the desired task. In the same way that today's Java programmer is relieved of low-level, machine-specific decisions (such as which data to keep in fast registers, and which in main memory or disk), the future crowdsourcing programmer would specify the goals of their system, and leave many of the implementation details to the Crowdsourcing Compiler. Such details might include which components of the task are best carried out by machine and which by human volunteers; whether the human volunteers should be incentivized by payment, recognition, or entertainment; how their contributions should be combined to solve

the overall task; and so on. We acknowledge the difficulty of this challenge --- at its logical extreme, the Crowdsourcing Compiler might be unattainable --- but at a minimum, significant progress towards it would imply a much deeper scientific understanding of crowdsourcing than we currently have, which in turn should have great engineering benefits.

We would note that the organizational schemes in virtually all of the successful crowdsourcing examples to date share much in common. The tasks to be performed (e.g. building an online encyclopedia, labeling images for their content, creating a network of website bookmark labels, finding surveillance balloons) are obviously parallelizable, and furthermore the basic unit of human contribution required is extremely small (fix some punctuation, label an image, etc.). Furthermore, there is very little coordination required between the contributions, and often the task is very open-ended (we want an online encyclopedia, but it's fine if the article about Britney Spears is three times longer than the one on the Vietnam War). The presence of these commonalities is a source of optimism for the Crowdsourcing Compiler --- so far, there seems to be some shared structure to successful crowdsourcing that the compiler might codify. But are such commonalities present because they somehow delineate fundamental limitations on successful crowdsourcing --- or is simply because this is the "low-hanging fruit", and no one has tried more ambitious tasks or designs yet?

The Crowdsourcing Compiler is an intriguing long-term challenge, but not (yet) a short-term agenda for crowdsourcing research --- it is too underspecified, and there are too many intermediate questions for which we do not yet have answers, for us to start working on a compiler tomorrow. But we feel the goal of the Crowdsourcing Compiler can help shape and focus a more concrete research agenda. Examples of some of the first questions this research might address include the following:

- For a given set of assumptions about the volunteer force, and given the nature of the task, what is the best scheme for organizing the volunteers and their contributions? For instance, is it a "flat" scheme where all contributors are equal and their contributions are combined in some kind of majority vote fashion? Or is it more hierarchical, with proven and expert contributors given higher weight and harder subproblems? Which of these (or other) schemes should be used under what assumptions on the nature of the task?

- How can we design crowdsourced systems for solving tasks that are much more challenging and less "transactional" than what we currently see in the field --- for instance, complex problems where there are strong constraints and interdependencies between the contributions of different volunteers? There is indeed strong evidence that such tasks can be tackled with crowdsourcing, at least in moderate size --- over the past five years at Penn, we have conducted an extensive series of human-subject experiments in solving difficult, interdependent tasks such as graph coloring, often with surprising behavioral success.