Exploring Iterative and Parallel Human Computation Processes

Greg Little[1], Lydia B. Chilton[2], Max Goldman[1], Robert C. Miller[1]

[1]MIT CSAIL
{glittle, maxg, rcm}@mit.edu

[2]University of Washington
hmslydia@cs.washington.edu

ABSTRACT

Services like Amazon's Mechanical Turk have opened the door for exploration of processes that outsource computation to humans. These *human computation processes* hold tremendous potential to solve a variety of problems in novel and interesting ways. However, we are only just beginning to understand how to design such processes. This paper explores two basic approaches: one where workers work alone in parallel and one where workers iteratively build on each other's work. We present a series of experiments exploring tradeoffs between each approach in several problem domains: writing, brainstorming, and transcription. In each of our experiments, iteration increases the average quality of responses. The increase is statistically significant in writing and brainstorming. However, in brainstorming and transcription, it is not clear that iteration is the best overall approach, in part because both of these tasks benefit from a high variability of responses, which is more prevalent in the parallel process. Also, poor guesses in the transcription task can lead subsequent workers astray.

Categories and Subject Descriptors

H5.m. Information interfaces and presentation (e.g., HCI): Miscellaneous.

General Terms

Algorithms, Measurement, Performance, Design, Reliability, Experimentation.

Keywords

Human Computation, Mechanical Turk, crowd sourcing, collaborative writing, brainstorming, OCR

1. INTRODUCTION

Countless examples on the web demonstrate the power of having a system outsource computation tasks to human workers. Human computation arguably wrote the world's largest encyclopedia. We even used it to decipher the text in Figure 1. However, human computation processes are still not well understood. In order to make the most of this new technology, to make it more efficient, and apply it to more problems, we need to break it down and understand it better.

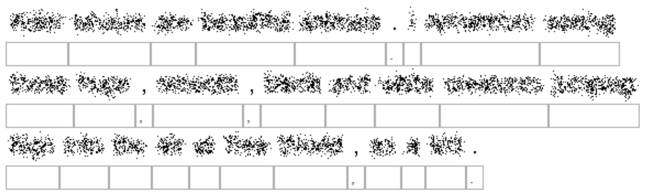


Figure 1: Mechanical Turk workers deciphered almost every word of this heavily blurred passage: *"Killer whales are beautiful animals. I remember seeing these huge, ~~beautiful~~, black and white creatures jumping high into the air at Sea World, as a kid."* ("beautiful" should be "smooth")

One difference we notice between human computation processes on the web is between iterative and parallel. For instance, if we look at how an article is written on Wikipedia, we often see an iterative process. One person starts an article, and then other people iteratively improve it by looking at what people did before them and adding information, correcting grammar, creating a consistent style, etc. On the other hand, designs on Threadless, a t-shirt design site, are generally created in parallel. Many people submit ideas independently, and then people vote to determine the best ideas that will be printed on a t-shirt.

We are interested in understanding the tradeoffs between each approach. The key difference seems to be that the iterative process shows each worker the results from previous workers, whereas the parallel processes asks each worker to solve a problem alone. The parallel process can be parallelized because no workers depend on the results of other workers, whereas the iterative process must solicit contributions serially.

To study the tradeoffs between each approach, we create a simple model wherein we can formally express the iterative and parallel processes. We then run a series of controlled experiments, applying each process to three diverse problem domains: writing image descriptions, brainstorming company names, and transcribing blurry text. These experiments are run using Amazon's Mechanical Turk as a source of on-demand labor. This allows us to run the experiments multiple times, increasing statistical validity. Each process is coordinated automatically using TurKit [11].

Our results show that iteration improves average response quality in the writing and brainstorming domains. We observe an increase in the transcription domain as well, but it is not statistically significant, and may be due to other factors. In the case of writing, the increase in quality comes from the ability of workers to collaboratively write longer descriptions. We also observe a few downsides for iteration. In the case of brainstorming, workers riff on good ideas that they see to create other good names, but the very best names seem to be generated by workers working alone. In transcription, showing workers guesses from other workers can lead them astray, especially if the guesses are self-consistent, but wrong.

Overall, prototyping and testing these processes on Mechanical Turk provides insights that can inform the design of new processes. The contributions of this paper include:

- Model and design patterns for human computation processes that coordinate small paid contributions from many humans to achieve larger goals.
- A series of experiments that compare the efficacy of parallel and iterative design patterns in a variety of problem domains.
- A discussion of the tradeoffs observed between the iterative and parallel approaches.

This paper now proceeds with a discussion of related work. Then we present our model of iterative and parallel processes, and apply each process to three problem domains, discussing the results of each in turn. We end with a generalized discussion of our results, as well as proposals for future work.

2. RELATED WORK

Human computation in general has recently received a lot of attention. Quinn and Bederson give a good overview of distributed human computation systems [14]. Individual systems have also been studied and explored in academic literature, including Games with a Purpose [1] [2] [3], Wikipedia [5] [9], and Mechanical Turk [7] [8] [13] [15] [16].

Researchers have also tried to break down and categorize human computation, including Quinn and Bederson [14]. Their *aggregation* dimension is most applicable to this paper. It asks how work is coordinated and combined to achieve a final result. Malone et al. [12] also break down collective intelligence systems along several dimensions. They make a distinction between workers working *dependently* and *independently*, which roughly maps to our iterative and parallel processes. They also distinguish between *creation* and *decision* tasks. For example, Threadless uses a creation task to generate a bunch of designs, and then a decision task to pick a design. We borrow these names, but apply them at a lower level, e.g., a creation task for us refers to a single worker generating a single design. This distinction is also made by Kosorukoff [10] using the names *innovation* and *selection*. Selection in this case comes from thinking of human computation as a genetic algorithm, which is an applicable analogy to both our iterative and parallel processes. However, we prefer a model that allows for the expression of human computation that does not resemble a genetic algorithm. Our model treats human computation as a set of operators in addition to traditional computation.

Iterative and parallel human computation processes have been implemented in a number of places on the internet. Iterative processes are seen in wikis and open source collaborations. Parallel processes are seen in contest sites like Threadless and news aggregation sites like Slashdot, reddit, and Digg, as well as sites that collect content, like YouTube. However, all of these sites are quite different, so it is hard to see a controlled comparison between iterative and parallel processes just by observing the web.

3. MODEL

We are interested in human computation processes which coordinate small contributions from many humans to achieve larger goals. For example, an algorithm might coordinate many workers to write a description for an image. All of the creative and problem solving power in these processes will come from humans, e.g., humans will do the writing.

Typical user generated content, like image descriptions, comes in a form that the computer does not understand. In order for the human computation process to make decisions, it will need to ask humans to evaluate, rate, or compare content such that the result is a number, boolean, or other data representation that a computer can readily process. This suggests a breakdown of the domain into *creation* and *decision* tasks.

3.1 Creation Tasks

When a worker writes a description for an image, this is a creation task. Creation tasks solicit new content: writing, ideas, imagery, solutions. Tasks of this form tend to have few constraints on worker inputs to the system, in part because the computer doesn't understand the input. The goal of a creation task is to produce new high quality content, e.g., a well written and informative description for an image.

Many factors affect the quality of results. We are interested in exploring the potential benefits of iteration, where each worker is shown content generated by previous workers. The hope is that this content will serve as inspiration for workers, and ultimately increase the quality of their results.

3.2 Decision Tasks

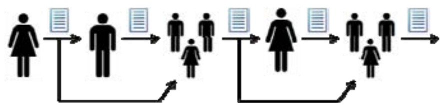
If we have two descriptions for the same image, we can use a decision task to let the process know which is best. Decision tasks solicit opinions about existing content. Tasks of this form tend to impose greater constraints on worker contributions, since the computer will need to understand the contributions. The goal of a decision task is to solicit an accurate response. Toward this end, decision tasks may ask for multiple responses, and use the aggregate. In our experiments, we use two decision tasks: comparison and rating. The comparison task shows a worker two items, and asks them to select the item of higher quality. The order of items is always randomized in our comparison tasks.

The rating task shows a worker some content, and asks them to rate the quality of the content, with respect to some goal, e.g., "Please rate the quality of this text as a description of the image." All rating tasks in this paper use a rating scale from 1 to 10.

3.3 Combining Tasks: Iterative and Parallel

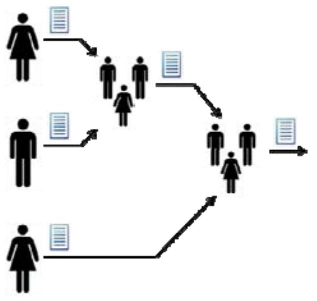
Human computation processes combine basic tasks in certain patterns. At this point, the patterns we explore are quite simple. All of the processes in this paper follow one of two patterns: iterative or parallel.

The iterative pattern consists of a sequence of creation tasks, where the result of each task feeds into the next one. The goal is to explore the benefits of showing workers content generated by previous workers. If it is not easy to merge the results of creation tasks, as is the case for image descriptions, then we place a comparison task between creation tasks:



This comparison lets the computer make a decision about which content to feed into the next task. In our experiments, we want to keep track of the best item created so far, and feed that into each new creation task.

The parallel pattern consists of a set of creation tasks executed in parallel. The parallel pattern acts as a control in our experiments—it is effectively the same as the iterative pattern, except that no workers are shown any work created by others. If it is not easy to automatically merge the results, then after all the results have been generated, we employ a sequence of comparison tasks to find the best result:



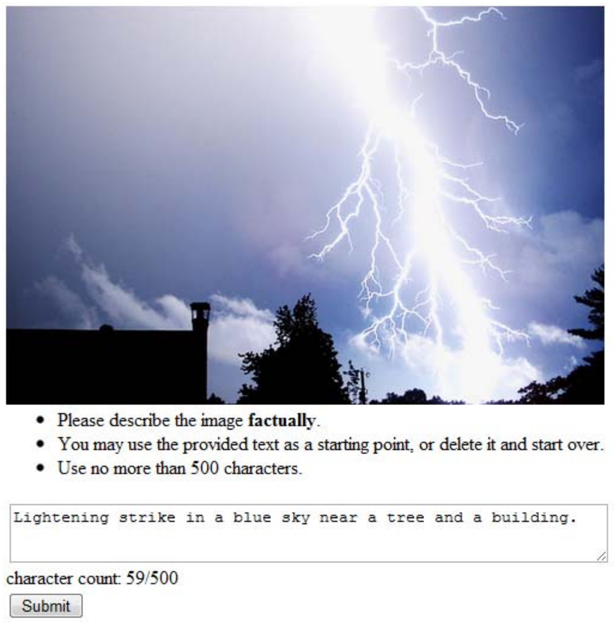
Our experiments also feed all subjective contributions into rating tasks in order to compare the effectiveness of each process.

4. EXPERIMENTS

In this section, we describe three experiments run on Mechanical Turk which compare the parallel and iterative processes in three different problem domains: writing, brainstorming, and transcription. Each domain was chosen because it solves a useful problem, and we wanted the combination to span a number of problem solving dimensions. For instance, the problems impose different levels of constraints: the transcription task has a definite answer, whereas the writing and brainstorming tasks are more open ended. The domains also vary in the size of each unit of work, ranging from single words to entire paragraphs.

4.1 Writing Image Descriptions

This experiment compares the parallel and iterative processes in the context of writing image descriptions. The experiment is inspired by Phetch [3], a game where humans write and validate image descriptions in order to make images on the web more accessible to people who are blind. This is a different approach to



- Please describe the image **factually**.
- You may use the provided text as a starting point, or delete it and start over.
- Use no more than 500 characters.

```
Lightening strike in a blue sky near a tree and a building.
```

character count: 59/500

Submit

Iterative: This image shows a large white strike of lightning coming down from a blue sky with the silhouettes of tops of the trees and rooftop peeking from the bottom. The sky is a dark blue and the lightening is a contrasting bright white. The lightening has many arms of electricity coming off of it. *rated 8.7*

Parallel: White lightning n [sic] a root-like formation shown against a slightly wispy clouded, blue sky, flashing from top to bottom. Bottom fifth of image shows silhouette of trees and a building. *rated 7.2*

Figure 2: Turkers are asked to write a factual description of an image. Turkers in the iterative condition are shown the best description so far, while the parallel condition always shows an empty text area. The resulting descriptions from each process are shown for this image.

the same problem that may be applicable to a greater variety of writing domains.

Each process has six creation tasks, each paying 2 cents. Five comparison tasks are used in each process to evaluate the winning description. Each comparison task solicits five votes, each for 1 cent. The instructions for the creation tasks are shown in Figure 2. The task asks a Mechanical Turk worker (*turker*) to describe the image factually in at most 500 characters. A character counter is updated continuously as the user types. The "Submit" button only activates when the content of the text area has changed from the initial text and there are at most 500 characters. Note that the instruction about "using the provided text" appears only in creation tasks that have text from a previous iteration to show. This instruction is omitted in all the parallel tasks.

To compare the processes, we selected 30 images from www.publicdomainpictures.net. Images were selected based on having interesting content, i.e., something to describe. We then ran both the parallel and iterative process on each image. For half of the images, we ran the parallel process first, and for the other half, we ran the iterative process first.

In order to compare the results from the two processes, we created a rating task. Turkers were shown an image and a description, and asked to rate the quality of the description as a factual description of the image, on a scale of 1 to 10. We obtained 10 ratings for each image description to compute an average rating.

Turkers were not allowed to participate in both processes for a single image. They were also not allowed to rate descriptions for images that they contributed any writing to. However, turkers were allowed to contribute to multiple images, as well as rate multiple descriptions for the same image.

Our hypothesis was that the iterative process would produce better results. We reasoned that workers would be willing to spend a constant amount of time writing a description, and they could do more with that time if they had a description to start from.

4.1.1 Results & Discussion

Figure 2 shows an example image, along with the resulting description for both the iterative and parallel processes. In this case, the iterative description is rated higher than the parallel description. If we average the ratings of resulting descriptions in each process for all 30 images, we get a small but statistically significant difference in favor of iteration (7.9 vs. 7.4, paired t-test $T_{29} = 2.1$, $p = 0.04$). Figure 3 shows what the result would have been if we had run the process for *n* iterations. Note that the two processes are identical when we use only one iteration.

It is worth noting that there is a correlation between description length and rating: longer descriptions are given higher ratings, accounting for about 30% of the variability in ratings according to the linear regression in Figure 4 ($R^2 = 0.2981$, $N = 360$, $\beta = 0.005$, $p < 0.0001$). This makes sense since we asked for "factual descriptions," and longer descriptions can hold more facts and details. The two circled outliers indicate cases of text copied from the internet that was only superficially related to the image.

This correlation is relevant because the iterative process produces longer descriptions, about 336 characters on average compared with 241 characters in the parallel process. This difference is enough to explain the difference we see in ratings. If we subtract the ratings predicted by a linear model of description length, then we are left with a residual rating of 0.37 for the iterative process, and 0.39 for the parallel process. However, a t-test reveals no statistically significant difference between these residual ratings ($p = 0.94$), meaning that the difference we saw before was probably due to description length.

However, note that the residual ratings are statistically significantly positive ($p = 0.014$), suggesting that each process produces descriptions that are rated higher than one might predict based on the length alone. This may be attributable to the fact that we have people vote between paragraphs, rather than simply comparing their lengths.

One simple model for what is happening in the iterative process is that it starts with a description of a certain length, and then subsequent turkers add more content. On average, turkers add about 25 characters in each iteration after the initial description. However, the standard deviation is very large (160), suggesting that turkers often remove characters as well. If we look more

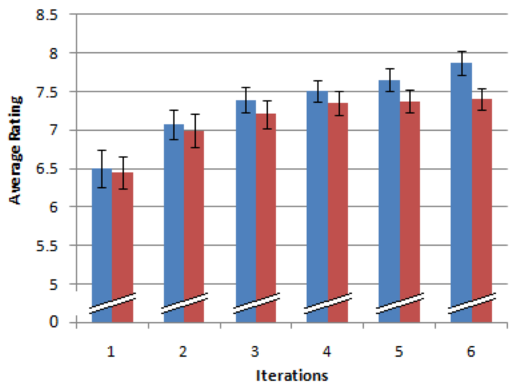


Figure 3: The average image description rating after *n* iterations (iterative process blue, and parallel process red). Error bars show standard error. As we run each process for additional iterations, the gap between the two seems to enlarge in favor of iteration, where the gap is statistically significant after six iterations (7.9 vs. 7.4, paired t-test T_{29} = 2.1, *p* = 0.04).

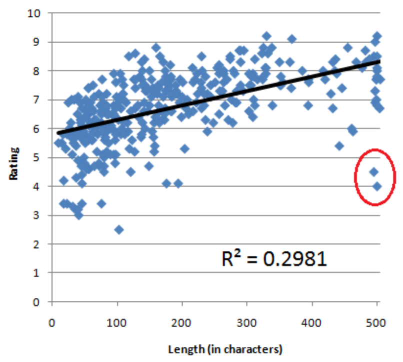


Figure 4: Descriptions are plotted according to their length and rating. A linear regression shows a positive correlation (R^2 = 0.2981, *N* = 360, β = 0.005, p < 0.0001). The two circled outliers represent instances of text copied from the internet.

closely at each of these instances, we can roughly classify their modifications as follows:

- 31% mainly append content at the end, and make only minor modifications (if any) to existing content;
- 27% modify/expand existing content, but it is evident that they use the provided description as a basis;
- 17% seem to ignore the provided description entirely and start over;
- 13% mostly trim or remove content;
- 11% make very small changes (adding a word, fixing a misspelling, etc);
- and 1% copy-paste superficially related content found on the internet.

Note that most modifications (83%) keep some of the existing content and structure, suggesting that these turkers may be doing less work. However, the average time spent by turkers writing or improving descriptions in each process is about the same, 211 seconds.

4.2 Brainstorming

This experiment compares the iterative and parallel processes in a different domain, namely brainstorming company names. Brainstorming is a popular process whereby many people generate ideas, either individually, or in a group. This process is well studied, and Taylor, et. al. [17] suggest that combining the results of individual brainstorms is more effective than having people brainstorm in a group. While group brainstorms typically generate fewer unique names, we try to mitigate this effect by programmatically enforcing that each worker contribute the same number of unique names in each process.

Each process has six creation tasks, each paying 2 cents. The instructions for these tasks are shown in Figure 5. The instructions ask a worker to generate five new company name ideas based on the provided company description. The "Submit" button only becomes active when there is text in each of the five input fields. The section that lists "Names suggested so far" only exists in the iterative condition. This list contains all names suggested in all previous iterations for a given company.

We fabricated descriptions for six companies. We then ran both the iterative and parallel process on each company description. As with the previous experiment, we ran the parallel variation first for half of the companies, and the iterative first for the other half. No turkers were allowed to contribute to both the iterative and parallel process of a single company description.

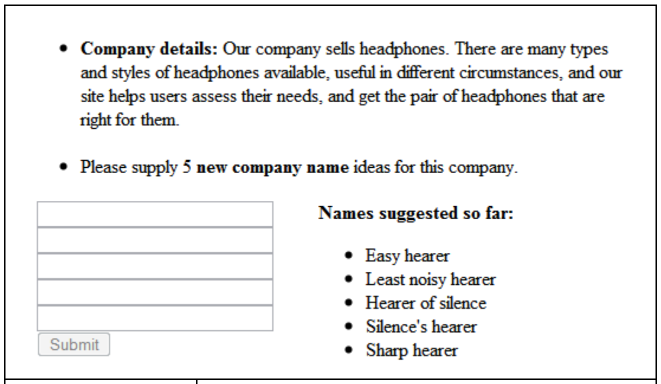
In order to compare the results of these processes, we used the rating technique discussed in the previous experiment to rate each generated company name. Again, we solicited 10 ratings for each company name, and averaged the ratings. Our hypothesis was that the iterative process would produce higher quality company names, since turkers could see the names suggested by other people, and build on their ideas.

4.3 Results & Discussion

By coincidence, no turkers in the parallel condition suggested duplicate names, resulting in 180 unique names for this condition. We removed duplicate names in the iterative condition from our data. They could have been prevented in JavaScript, and 13 of the 14 duplicate names came from the last three iterations for a single company. This may have been due to a group of turkers working together that collectively misunderstood the directions, or tried to cheat (unfortunately we did not record IP addresses, so we do not know if these turkers were collocated).

Figure 5 shows a fake company description, along with a sorted sample of the names suggested for this company. The best rated name generated in the iterative process is rated 7.3, compared with 8.3 for the parallel process. In fact, the parallel process generated the best rated name for 4 out of the 6 fake companies.

However, the *average* name generated in the iterative process is rated higher (6.4 vs. 6.2). The significance of iteration becomes clear in Figure 6, where we show the average rating of names generated in each iteration of the iterative process. The red line indicates the average rating of names in the parallel process. The iterative process is close to this line in the first iteration, where turkers are not shown any example names. The average rating seems to steadily increase as turkers are shown more and more examples (except for iteration four, discussed next). The last iteration is statistically significantly higher than the parallel process (6.7 vs. 6.2, two-sample $T_{203} = 2.3$, $p = 0.02$).



- **Company details:** Our company sells headphones. There are many types and styles of headphones available, useful in different circumstances, and our site helps users assess their needs, and get the pair of headphones that are right for them.
- Please supply 5 **new company name** ideas for this company.

Submit

Names suggested so far:

- Easy hearer
- Least noisy hearer
- Hearer of silence
- Silence's hearer
- Sharp hearer

Iterative:	• Easy on the Ears *rated 7.3* • Easy Listening *7.1* • Music Explorer *7.1* • Right Choice Headphone *7.1* … • Least noisy hearer *5.1* • Headphony *4.9* • Shop Headphone *4.8*
Parallel:	• music brain *rated 8.3* • Headphone House *7.4* • Headshop *7* • Talkie *6.8* … • company sell *4.3* • head phones r us *4.2* • different circumstances *3.7*

Figure 5: Turkers are asked to generate five new company names given the company description. Turkers in the iterative condition are shown names suggested so far. The highest and lowest rated names from both the iterative and parallel processes are shown for this company.

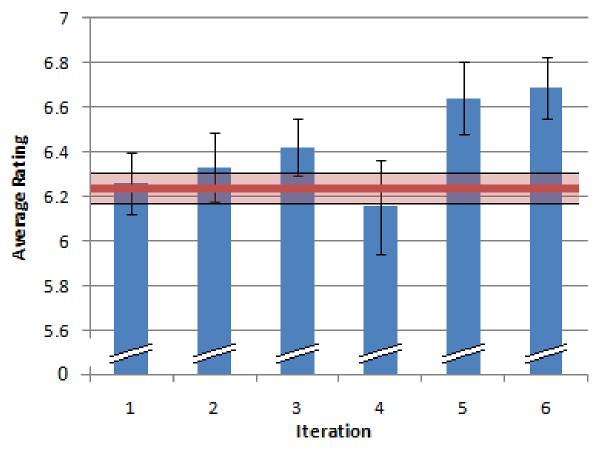


Figure 6: Blue bars show average ratings given to names generated in each of the six iterations of the iterative brainstorming processes. Error bars show standard error. The red stripe indicates the average rating and standard error of names generated in the parallel brainstorming processes. (See the text for a discussion of iteration 4, which appears below the red line.)

Iteration four breaks the pattern, but this appears to be a coincidence. Three of the contributions in this iteration were considerably below average. Two of these contributions were made by the same turker (for different companies). A number of their suggestions appear to have been marked down for being grammatically awkward: "How to Work Computer", and "Shop Headphone". The other turker suggested names that could be considered offensive: "the galloping coed" and "stick a fork in me".

Like the previous task, we did not observe a statistically significantly difference in the time that turkers spent generating names in each process: 202 seconds for iterative, and 178 seconds for parallel (two-sample $T_{70} = 1.1$, $p = 0.28$).

4.3.1 Getting the Best Names

Although iteration seems to increase the average rating of new names, it is not clear that iteration is the right choice for generating the *best* rated names (recall that the parallel process generated the best rated name for 4 of the 6 fake companies). This may be because the iterative process has a lower variance: 0.68 compared with 0.9 for the parallel process (F-test, $F_{180,165} = 1.32$, $p = 0.036$). The higher variance of the parallel distribution means that the tail of the distribution does not shrink as quickly, and must eventually surpass the iterative process. Figure 7 illustrates this phenomenon, and shows the crossover point at a rating of 8.04. This suggests that the parallel process is more likely than the iterative process to generate names rated 8.04 and above, assuming that the names are normally distributed according to this model.

Note that this model may not tell the whole story. The maximum possible rating is 10, which suggests that Gaussian curves are not the best model (since they do not have any bounds). A Beta distribution may be more appropriate. This does not mean that the effect we are seeing isn't real, but it is worth further investigation.

One high level interpretation of this is that showing turkers suggestions may cause them to riff on the best ideas they see, but makes them unlikely to think too far afield from those ideas. We did see some anecdotal evidence of people's ideas being heavily influenced by suggested names. For an online chat company, the first turker suggested five names that all incorporated the word "chat". The next two turkers also supplied names that all incorporated the word "chat". The corresponding iterations in the parallel process only used the word chat three out of fifteen times. For another company, a turker used the word "tech" in all their names, and subsequent turkers used the word "tech" seven times, compared with only once in the corresponding parallel iterations.

This suggests some interesting questions to explore in future work: does showing people name suggestions inhibit their ability to generate the very best new name ideas? If so, is there anything we can do in an iterative process that *will* help generate the best names? Alternatively, is there any way we can increase the variance in the parallel process, to have an even better chance of generating the best names?

4.4 Blurry Text Recognition

This experiment compares the iterative and parallel processes in the transcription domain. The task is essentially human OCR, inspired by reCAPTCHA [4]. We considered other puzzle possibilities, but were concerned that they might be too fun, which could have the side effects discussed in [13]. Unlike the

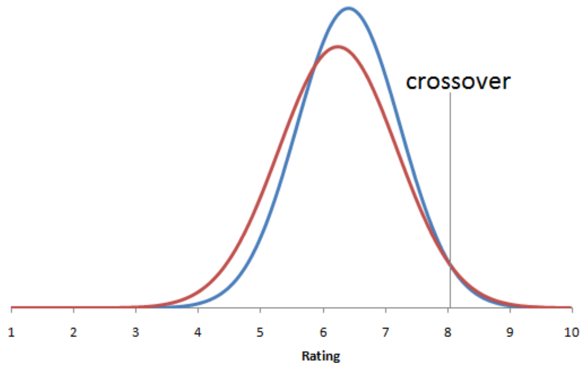
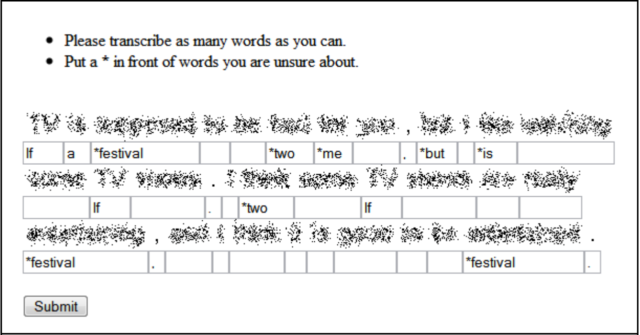


Figure 7: Gaussian distributions modeling the probability of generating names with various ratings in the iterative (blue) and parallel (red) processes. The mean and variance of each curve is estimated from the data. The iterative process has a higher average, but the parallel process has more variance (i.e. the curve is shorter and wider). Note that in this model, the parallel distribution has a higher probability of generating names rated over 8.04.



- Please transcribe as many words as you can.
- Put a * in front of words you are unsure about.

If | a | *festival | *two | *me | . | *but | *is

If | . | *two | If

*festival | . | *festival | .

Iterative: TV is supposed to be bad for you, but I ~~am~~ watching some TV shows. I think some TV shows are really entertaining, and I think it is good to be ~~watched~~. *(94% correct)*

Parallel: TV is supposed to be bad for you, but I like watching some TV shows. I think some TV shows are really ~~advertising~~, and I think it is good to be entertained. *(97% correct)*

Figure 8: Turkers are shown a passage of blurry text with a textbox beneath each word. Turkers in the iterative condition are shown guesses made for each word from previous turkers. The resulting transcription from each process is shown, with incorrect words struck out in red.

previous experiments, we use sixteen creation tasks in both the iterative and parallel processes, each task paying 5 cents.

Figure 8 shows an example blurry text recognition task. The instructions are simply to transcribe as many words as possible, and we place a textbox beneath each word for this purpose. In the iterative condition, these textboxes contain the most recent guess for each word. We also ask workers to put a '*' in front of words that they are unsure about. This is meant as a cue to future workers that a word requires more attention. Note that this instruction appears in the parallel tasks as well, even though no workers see any other workers' stars.

We composed twelve original passages. It was important to use original text, rather than text obtained from the web, since turkers could conceivably find those passages by searching with some of the keywords they had deciphered.

We then ran each passage through an image filter. The filter works on a pixel level. Each pixel in the new image is created by randomly choosing a pixel near that location in the old image, according to a 2-dimensional Gaussian. This is identical to the lossy "blur" tool from some popular image editing programs. The result is blurry text that is very difficult to decipher. Some words appear to be entirely illegible on their own. The hope is that by seeing the entire passage in context, turkers will be able to work out the words. Note that we don't need to pay any workers to rate the results, since we can assess the accuracy of the results automatically using the ground truth text from which the blurry images were created.

We applied both the iterative and parallel process to each passage. As before, each process was run first half the time, and no turkers were allowed to participate in both processes for a single passage. The final transcription of the blurry text is extracted from each process on a word by word basis. We look at all the guesses for a single word in all sixteen iterations. If a particular word is guessed a plurality of times, then we choose it. Otherwise, we pick randomly from all the words that tied for the plurality. Note that other algorithms are possible, and we will have more to say about this below.

Our hypothesis was that the iterative process would have a higher probability of deciphering each passage, since turkers would be able to use other people's guesses as context for their own guesses. The analogy would be solving a cross-word puzzle that other people have already started working on.

4.4.1 Results & Discussion

Figure 8 shows a passage, along with the transcription extracted from both the iterative and parallel processes. In this case, the parallel process does a slightly better job (97% of words transcribed correct vs. 94%). When we average over all 12 passages, we fail to see a statistically significant difference between each process (iterative 65% vs. parallel 62%, paired t-test $T_{11} = 0.4$, $p = 0.73$).

If we look at the accuracy of each process after *n* iterations in Figure 9, we see that both processes gain accuracy over the first eight iterations, and then seem to level off. Note that the iterative process appears to be above the parallel process pretty consistently after the fourth iteration. The difference is greatest after eight iterations, but is never statistically significant.

The results suggest that iteration may be helpful for this task. However, it is also worth noting that iteration sometimes appears to get stuck due to poor guesses early in the process. For instance, one iterative process ended up with 30% accuracy after sixteen iterations. The final result was very similar to the eighth iteration, where most of the words had guesses, and they made a kind of sense:

8th iteration: "Please do ~~ask~~ * anything ~~you need *me~~. Everything is ~~going fine~~, ~~there~~ * * , ~~show me then~~ * * anything you ~~desire~~."

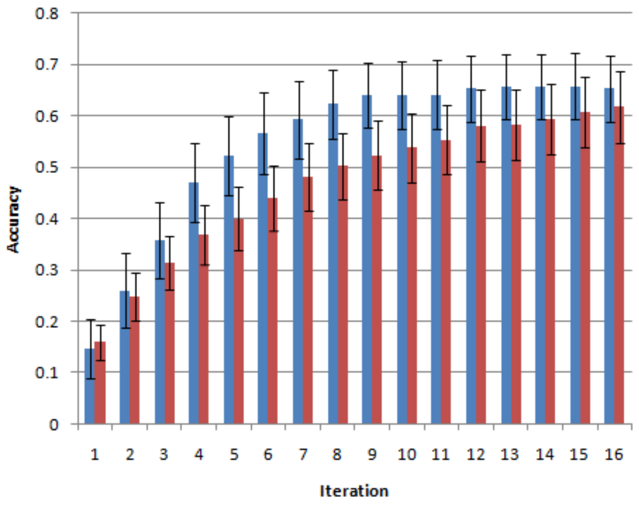


Figure 9: Blue bars show the accuracy after *n* iterations of the iterative text recognition process. Red bars show accuracy for the parallel process with *n* submissions. Error bars show standard error. The overlap suggests that the processes may not be different, or that we do not have enough power in our experiment to see a statistically significant difference.

16th iteration: "Please do ~~ask *about~~ anything ~~you need *me~~. Everything is ~~going fine, there *were~~ * , show me then ~~*bring~~ * anything you ~~desire~~."

Incorrect guesses have been crossed out in red. Here is the actual passage: "Please do not touch anything in this house. Everything is very old, and very expensive, and you will probably break anything you touch." Note that multiple turkers deciphered this entire passage almost perfectly in the parallel process, suggesting that progress was hampered by poor guesses rather than by unreadable text. In fact, one turker left a comment alluding to this possibility (but for a different passage): "It's distracting with the words filled in--it's hard to see anything else but the words already there, so I don't feel I can really give a good "guess" ".

Note that in this experiment, we did see a statistically significant difference in the time spent deciphering text. Turkers in the iterative task spent less time, an average of 130 seconds, compared with 159 seconds in the parallel task (two-sample $T_{382} = -2.03$, $p = 0.043$).

4.4.2 Extraction Algorithm

Although iteration appears to be marginally better, some of the benefit may have to do with the algorithm we use to extract a final answer from all the guesses. If we had a perfect algorithm, one that could look at all the guesses for a particular word and choose the correct guess if it existed, even if it was not the most common guess, then the parallel process would do better: 76% versus 71% for the iterative process. This difference is not significant (p = 0.64), and only serves to cast greater doubt on the benefit of iteration for this task.

This suggests that any benefit we see from iteration may have to do with the fact that it has an implicit mechanism for multiple turkers to vote on a single turker's guess. If a turker leaves a guess alone, then they are implicitly voting for it. In the parallel process, a word can only get multiple votes if multiple turkers arrive at the same guess independently.

One way the parallel process could be improved would be using other information to help pick out the best guesses. For instance, a word may be more likely to be correct if it came from a turker who made guesses for surrounding words as well, since the word may be satisfying more constraints.

It may be possible to improve the iterative process as well by randomly hiding guesses for certain words, in order to solicit more options for those words. Also, we could show people multiple options for a word, rather than just the most recent guess.

Although it is interesting to think of ways to improve the algorithm for this process, our real hope was that this task would be so difficult that a single turker could not accomplish it on their own. In future work, it would be nice to explore a task that had a higher level of difficulty, or impose a time limit to simulate higher difficulty.

5. DISCUSSION

All of these experiments demonstrate success in performing several relatively high-level creative and problem solving tasks, using processes that orchestrate the efforts of multiple workers. We also see that the breakdown into creation and decision tasks is applicable to a diverse set of problem domains.

5.1 Tradeoff between Average and Best

In the brainstorming task, we saw a tradeoff between increasing average response quality, and increasing the probability of the best responses. Showing prior work increased the average quality of responses, but reduced the variance enough that the highest quality responses were still more likely to come from turkers not shown any prior suggestions.

There is a sense in which this tradeoff exists in the other two tasks as well. The writing task generates descriptions with a higher average rating, but lower variance, when turkers are shown prior work. Also, the transcription task increases the average frequency of correct guesses for each word when turkers are shown a prior guess, but it decreases the variety of guesses for each word.

However, an alternate explanation for the reduced variance is simply that there is a maximum quality, so pushing the average toward this barrier must eventually reduce the variance (e.g., if the average rating reaches 10, then the variance must be 0). The real question is whether the reduction in variance is enough to exhibit the tradeoff we seem to observe in the brainstorming case. We will need to employ more robust mathematical models to make more progress on this front, since Gaussian distributions are limited in how well they can model a bounded random variable.

Investigating this tradeoff further may be worthwhile, because if it exists, then it implies two different alternatives for achieving quality responses, depending on our target quality. If our target quality is reasonably low, then we can increase the average, whereas if it is very high, then we may do better to increase the variance and find some method of detecting the high quality responses when they arrive.

Note that we may be able to proactively increase the variance in the brainstorming task, perhaps by showing people completely random words (which may have nothing to do with the topic), in order to get them thinking outside the proverbial box.

5.2 Not Leading Turkers Astray

In our experiments, showing prior work can have a negative effect on quality by leading future workers down the wrong path. This effect is most pronounced in the blurry text recognition task, and may be an issue in other tasks of this form where puzzle elements build on each other (like words in a crossword puzzle). Turkers take suggestions from previous turkers, and try to make the best guesses they can, but backtracking seems more rare.

This suggests that a hybrid approach may be better, where multiple iterative processes are executed in parallel to start with, and then further iteration is performed on the best branch.

6. FUTURE WORK

There are many directions for future work exploring human computation processes on Mechanical Turk. First, it seems fruitful to further explore the two basic building blocks: creation tasks and decision tasks. For instance, there are many factors that may influence creation tasks, including price, how much work is expected, whether examples are shown, and whether prior work is shown. We have only scratched the surface of exploring these possibilities, but as these different elements are better understood, it will be easier to design creation tasks with the desired balance between the average quality and variance of responses.

Decision tasks also leave room for investigation. The basic goal in our tasks has usually been to determine the best items in a set, but there are a number of ways to achieve this, including absolute ratings, pair-wise comparisons, and sorting multiple items in a single task. There are even combinations of these elements that may be useful, like having people sort items, but also provide ratings. Again, this is a large space, with the promise of providing useful knowledge to help optimize evaluation of subjective content.

Another direction for future work is exploring new building blocks. Even inside the paradigm of creation and decision tasks, there is room for building blocks which sit somewhere in-between. For instance, a creation writing task could ask a turker to select which of two previous versions they want to start from (where they are effectively voting for which of those is best). Note that this could have the side effect of turkers selecting the worse version from which to start, since it may be easier to improve. Hence, these ideas need to be tested and validated.

Finally, the real creative potential in this space is exploring new high-level processes for coordinating workers to perform better, or achieve loftier objectives. For instance, in paragraph writing, one can imagine breaking down the task into two steps. The first step might have people brainstorm phrases or concepts that should be included in a paragraph, and then these could be shown to the people writing the actual paragraph.

One might also imagine writing something larger than a paragraph through several steps: writing an outline, having separate processes to write each paragraph from the outline, and having another process to combine these results with transition sentences into a complete essay.

The ultimate goal is to learn how to design processes on Mechanical Turk that reliably and efficiently achieve their objectives, and to push the boundaries of those objectives.

7. CONCLUSION

This paper compares iterative and parallel human computation processes. In the iterative algorithm, each worker sees the results from the previous worker. In the parallel process, workers work alone. We apply each algorithm to a variety of problem domains, including writing, brainstorming, and transcription. We use Mechanical Turk and TurKit to run several instances of each process in each domain. We discover that iteration increases the average quality of responses in the writing and brainstorming domains, but that the best results in the brainstorming and transcription domains may come from the parallel process, because it yields a greater variety of responses. We also see that providing guesses for words in the transcription domain can lead workers down the wrong path. These results provide insights that can inform the design of new human computation processes.

8. ACKNOWLEDGMENTS

We would like to thank everyone who contributed to this work, including Mark Ackerman, Michael Bernstein, Thomas W. Malone, Robert Laubacher, and members of the UID group. This work was supported in part by Xerox, by the National Science Foundation under award number IIS-0447800, by Quanta Computer as part of the TParty project, and by the MIT Center for Collective Intelligence. Any opinions, findings, conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of the sponsors.

9. REFERENCES

[1] von Ahn, L. Games With A Purpose. IEEE Computer Magazine, June 2006. Pages 96-98.

[2] von Ahn, L., and Dabbish, L. Labeling Images with a Computer Game. ACM Conference on Human Factors in Computing Systems, CHI 2004. Pages 319-326.

[3] von Ahn, L., Ginosar, S., Kedia, M., and Blum, M. Improving Accessibility of the Web with a Computer Game. ACM Conference on Human Factors in Computing Systems, CHI Notes 2006. pp 79-82.

[4] von Ahn, L., Maurer, B., McMillen, C., Abraham, D. and Blum, M. reCAPTCHA: Human-Based Character Recognition via Web Security Measures. Science, September 12, 2008. pp 1465-1468.

[5] Bryant, S. L., Forte, A. and Bruckman, A.. Becoming Wikipedian: transformation of participation in a collaborative online encyclopedia. GROUP 2005.

[6] Dai, P., Mausam, Weld, D. S. Decision-Theoretic Control of Crowd-Sourced Workflows. AAAI 2010.

[7] Heer, J., Bostock, M. Crowdsourcing Graphical Perception: Using Mechanical Turk to Assess Visualization Design. CHI 2010.

[8] Kittur, A., Chi, E. H., and Suh, B. 2008. Crowdsourcing user studies with MTurk. CHI 2008.

[9] Kittur, A. and Kraut, R. E. 2008. Harnessing the wisdom of crowds in wikipedia: quality through coordination. CSCW '08. ACM, New York, NY, 37-46

[10] Kosorukoff A. Human based genetic algorithm. IlliGAL report no. 2001004. 2001, University of Illinois, Urbana-Champaign.

[11] Little, G., Chilton, L. B., Goldman, M., and Miller, R. C. TurKit: Tools for Iterative Tasks on Mechanical Turk. HCOMP 2009.

[12] Malone, T. W., Laubacher, R. and Dellarocas, C. Harnessing Crowds: Mapping the Genome of Collective Intelligence. MIT, Cambridge, 2009.

[13] Mason, W., Watts, D. J. Financial Incentives and the "Performance of Crowds". HCOMP 2009.

[14] Quinn, A. J., Bederson, B. B. A Taxonomy of Distributed Human Computation. Technical Report HCIL-2009-23 (University of Maryland, College Park, 2009).

[15] Snow, R., O'Connor, B., Jurafsky, D., and Ng, A. Y. Cheap and fast---but is it good?: evaluating non-expert annotations for natural language tasks. EMNLP 2008.

[16] Sorokin, A. and D. Forsyth. Utility data annotation with Amazon MTurk. Computer Vision and Pattern Recognition Workshops, Jan 2008.

[17] Taylor, D. W., Berry, P. C. and Block, C. H. Does Group Participation When Using Brainstorming Facilitate or Inhibit Creative Thinking? Administrative Science Quarterly, Vol. 3, No. 1 (Jun., 1958), pp. 23-47.