Using Crowd Sourcing to Measure the Effects of System Response Delays on User Engagement

Brandon Taylor
Carnegie Mellon University
bttaylor@cs.cmu.edu

Anind K. Dey
Carnegie Mellon University
anind@cs.cmu.edu

Daniel Siewiorek
Carnegie Mellon University
dps@cs.cmu.edu

Asim Smailagic
Carnegie Mellon University
asim@cs.cmu.edu

ABSTRACT
It is well established that delays in system response time negatively impact productivity, error rates and user satisfaction. What is less clear is the degree to which these effects deter users from engaging with a system. Usability guidelines provide rough response time targets for minimizing these effects across various types of interactions. However, developers faced with technical limitations or cost constraints that prevent them from meeting such targets are given no data with which to estimate the impact that system response delays will have on user engagement. In this work, we demonstrate a methodology for using crowd sourcing platforms to examine (1) the relative impacts of different delay types and (2) the effects of marginal changes in system response times. We compare two common network delay types, those caused by limited bandwidth (increased download times) and those caused by network latency (lag in responsiveness), and present how these delays reduce engagement in the context of a crowd sourced image classification task. Furthermore, we model how financial incentives interact with system response delays to impact user engagement. Finally, we show how such models can be used to optimize the cost of system design choices.

Author Keywords
System Response Delays; Usability; Crowd Sourcing

ACM Classification Keywords

INTRODUCTION
System response delays are an unavoidable aspect of computer systems that have a dramatic impact on user experience. In an ideal world, system response delays would be reduced beyond perceptual thresholds when seamless interactions are needed, and introduced only when breaks in computational workflow would help to focus [5] a user's attention. In the real world, though, system response delays are often an unpredictable side effect produced by a combination of limited computational resources [4], greedy application processes [30], and the physical limits of information transfer.

Understanding how users are affected by system response delays has long been a topic of interest in HCI. Most work related to system response delays can be traced back to Robert Miller's seminal work on the matter in 1968 [21]. Following Miller's suggestions, human factors research put many of his assertions to the test and gained empirical results correlating system response delays with increases in error rates and decreases in productivity and user satisfaction [6,8,9,10,16,32,33].

Over the years, the findings from these studies have been codified into rules of thumb for interaction designers. Usability textbooks vary, but response time guidelines generally offer the advice that one or two seconds "seems appropriate for many tasks" [31], whereas response times under 0.1 second will seem as though "the system is reacting instantaneously" [24]. While these texts do make it clear that these heuristics do not apply in all cases, they do not offer much advice for determining specific system response time requirements for an application beyond "measuring productivity, errors, and the cost of providing short response times" [31].

While some researchers and developers have the resources and expertise to run full-scale user studies to find satisfactory limits of system response times, certainly not all do. As a result, situations arise where heuristics distilled from system response studies conducted on terminal text-entry systems are being used to estimate the usability of a graphics-intensive photo-editing task [18,34].

In this work, we present a methodology adapted from [35] for measuring both the effects of marginal changes in system response delays and the relative impacts of different sources of delays on user engagement. By using a crowd sourcing platform, we are able to easily collect user engagement data from thousands of individuals. We demonstrate how this methodology can be used to evaluate the impacts that limited bandwidth and high network latency have on user engagement. By separately manipulating financial incentives, we are also able to model the user engagement in response to financial costs. Following such an approach allows a developer to better estimate whether paying for increased bandwidth or

CHI'16, May 07-12, 2016, San Jose, CA, USA
 ISBN 978-1-4503-3362-7/16/05 $15.00
DOI: http://dx.doi.org/10.1145/2858036.2858572

additional servers to reduce delays would pay off in terms of improved user engagement.

The rest of this paper will be structured as follows. We will first address related work, covering both previous efforts to measure the impacts of system response delays and a methodology for using crowd sourcing platforms to measure user preferences. We will then describe the design of our study, covering the rationale behind our choices and the technical implementation details. Our findings will be separated into three sections. First we will compare how downloading delays, such as those caused by limited bandwidth, affect a user's willingness to engage with an application. Next, we will examine how users' motivation, adjusted by varying financial incentives, alters engagement. We will then discuss a variation on these studies that explores the impact of network latency instead of download delays. With these results, we will examine how financial incentives can impact tolerance for delays. Finally, we will discuss how researchers and practitioners can apply our methodology to better understand acceptable response delays in a variety of applications.

RELATED WORK

The impact of system response delays on user productivity and satisfaction was once an active area of human-computer interaction research. While many effects were measured, no model of user behavior has been able to fully account for the variations across users and contexts. In this section, we will first highlight the findings of prior work on system response delays. We will then examine the limitations of the methodologies used and discuss how these have limited the usefulness of usability guidelines derived from these studies. Finally, we will examine a crowd sourcing methodology that we have adapted for addressing these limitations.

Effects of System Response Delays

Decades of human factors research have examined how delays in system response impact user experience. Surveys have demonstrated that system response time is the most important factor related to user satisfaction with computer systems [29]. Work stemming from mainframe time-sharing studies has shown that performance decreases as system response time increases [6,8,9]. Stress has also been shown to increase along with system response time [16].

The emergence of the Internet and re-emergence of network latencies as a common source of delay has spurred more recent studies. These studies have demonstrated a negative correlation between response time and satisfaction with web pages and browser-based applications [11,23]. Individuals have been shown to find content less interesting [26] and to consider websites to be of lower quality [14] when response delays are increased.

Over the years, these and other studies have led to general guidelines of the 'acceptable' response time limits for various activities. As Nielsen described it, response times under 0.1 seconds will be treated as instantaneous, up to 1 second can be tolerated without breaking a user's train of thought and up to 10 seconds can go by without losing the user's attention [24]. For longer response times, research has shown that providing progress indicators is important for managing users' expectations [22]. However, while these heuristics provide rough *targets* for UX designers, they do not provide information about how to address the effects of smaller variations in delay that may arise or about the relative impacts of different types of delay.

System Response Delays in Context

Amongst the reasons that usability guidelines have not been able to provide developers with specific recommendations are the impacts of context and user expectations. In 1968, Robert Miller wrote perhaps the most influential work on response time requirements for 17 different contexts in which "different kinds of response and response delay will be appropriate" [21]. The contexts discussed ranged from the response to a key being struck ("no more than .1 to .2 seconds") to the execution of a program script (within 15 seconds to keep "in the problem-solving frame of mind").

In addition to these contexts, Miller also noted that "need is equivalent to what is demanded and what can be made available; need, therefore is a cultural and technical outcome". In Shneiderman's textbook on UI design, the importance of expectations are highlighted by the example of network managers who install network chokes so that users do not experience maximum connection speeds during low traffic periods. While this obviously introduces unnecessary delays, it also reduced complaints since users did not know what they were missing [31].

To add yet another wrinkle to the already difficult problem of understanding users' expectations, the rise of mobile devices have drastically increased the environmental contexts in which users may be operating. At the same time, fluctuations in mobile network resources only increase the variability of performance [30].

Given the multitude of factors that have been shown to impact a user's tolerance for system response delays, it is understandable why usability guidelines have produced limited heuristics. Modeling the effects of response delays is incredibly difficult without even considering how the models could be applied to new application development.

Crowd Sourced Measurements

An alternative to modeling user responses to system delays would be to instead *measure* the user responses. Crowd sourcing platforms have been used to explore a variety of research topics [15,25]. A recent study by Lasecki et al. demonstrated that both contextual shifts between tasks and temporal delays between tasks reduce task completion rates [19]. However, this study did not find a significant effect on task completion times for delays less than 30 seconds and did not explore how such delays affected a worker's willingness to continue working on a specific task.

Toomim et al. described the use of crowdsourcing platforms to examine worker choices and assign measures of utility to interfaces [35]. They first implement a simple Fitts' Law task in which users are asked to alternately click two buttons. Task difficulty was manipulated by controlling the size of the buttons. After each set of 60 clicks, users had the option to repeat the task (for additional payment) or to stop. They found that the users' willingness to continue was inversely proportional to the time to complete the task as predicted by Fitts' Law. Thus, they demonstrated that well-studied findings could be reproduced using this new methodology.

This approach was also used to demonstrate that crowd sourcing could be used to explore other effects as well. In another study, Toomim et al. manipulated the appearance of the web page hosting the tasks and were able to measure the impact of aesthetics on worker engagement. By repeating the comparisons with different levels of monetary reward, they were able to create a cost function with which to compare design choices.

STUDY DESIGN

Inspired by the example of Toomim et al.'s exploration of web page aesthetics, we sought to design a study that would allow us to measure the impacts of different types of network delays in a real-world setting. In particular, we wanted to compare the relative impacts of bandwidth limitations, commonly resulting in download delays, with network latency delays. Additionally, we hope to relate the impact of these delays on user engagement to monetary incentives. By doing so, developers will have an exemplar method for optimizing user engagement at set price points.

To achieve both of these goals, we chose a series of image classification tasks posted to Amazon's Mechanical Turk platform as a testbed. While we explored different delay types, the interface and execution were consistent across the studies. In this section we will touch on ethical concerns surrounding our study design then describe our interface and implementation details.

Observing Unbiased Behavior

A key aspect of our study was the need to observe worker behavior in response to variable delays without biasing their actions by drawing attention to the delays. In order to do so, we chose to present our study as a straightforward image recognition task. The task description simply stated, "Please look at a set of (5) images and select the best category and briefly describe the image (in a word or two)." No indication was given that delays were being introduced or that the responses would be used in any particular way.

Given the population of Mechanical Turk workers and their motivations for using the platform [13,28], this choice was not free of ethical concerns. By introducing delays, we were effectively reducing the number of tasks an individual could complete in a fixed time frame, thereby reducing their potential earnings. However, the Mechanical Turk website provides workers with a marketplace of available tasks at different price points, ostensibly allowing users to choose for themselves whether an individual task is worth completing (based on their own sense of value). In fact, our study was predicated on the idea that delay and price impact a worker's willingness to continue engaging in a task.

Additionally, the delays that we introduced were both substantially less than delays introduced in other studies [19] and comparable to delays naturally encountered in Wide-area Networks [2]. While the intentionality of this introduction and the withholding of this information may be questioned, we felt that it was necessary to achieve unbiased results. Furthermore, we feel that the scientific value of the study and the possibility that these findings will influence future task providers to better provision their networks outweighs the potential harms. Prior to running the study, an independent review board approved the study design with a waiver of informed consent.

Figure 1. Screenshot of an image identification task

Crowd Sourcing Task Design

We posted a series of Human Intelligence Tasks (HITs) to the Mechanical Turk website. Each series that we posted consisted of 50 unique HITs, with each HIT consisting of five image classification tasks. An example of an image classification task can be seen in Figure 1. Each HIT could be assigned to 100 workers, but a worker could only complete each unique HIT one time. Thus, every worker had the opportunity to complete a maximum of 50 sets of image classification tasks.

Each series of posts were made available for 24 hours and workers were rewarded for all completed HITs without any regard to their responses. Reward levels and delay types

were held constant across each series of HITs and no more than one series was available at any given time.

Upon completing a set of five images, the individual HIT would be complete. Workers could then choose to work on another of our HITs. For workers who chose to continue, a maximum of 50 HITs were available during any given posting.

When a user chose to work on one of our HITs for the first time, our server would record their Mechanical Turk ID and assign them a delay condition. This delay condition would be used for all subsequent tasks for that user. If the user had worked on a HIT during any previous series of postings, they would receive a message that no more image sets were available to them.

The image classification tasks consisted of a single html page and a set of JavaScript functions hosted on our server. Client-side functions received and implemented the delay conditions, verified the worker response completion, and recorded information about the worker's browser and timestamps upon page loads and button clicks. Server-side scripts recorded the information into our database.

Users were required to provide a short description of each image and select the most appropriate category from amongst four options. The images were obtained by searching for the category tags (e.g., 'Animal') on Flickr and manually verified to avoid any potential overlapping categories. The workers' category selection and description were preserved as a potential metric for filtering out malicious responses. However, 98.6% of responses were correctly categorized with no evidence of disingenuous responses for any individual worker.

Study 1: Download Delay

Studies have clearly demonstrated that increases in web page loading delays result in reduced satisfaction and more negative perceptions towards the web site [11, 14]. However, given any particular site and level of delay, there is no a priori way to predict the level of satisfaction or, more importantly, how reductions in satisfaction will drive visitors away. The first study we ran was designed to examine how increases in download delays reduce willing engagement with a site. Armed with such information, and with a methodology for replicating such a study in other application contexts, developers will be able to tailor their systems to optimize user engagement.

As described in the Study Design section, workers were assigned a delay condition of zero, two, or five seconds upon first choosing one our posted HITs. We originally had a fourth delay condition of 10s, but the results did differ significantly from the 5s delay case using a Log-rank test [17]. To conserve our resources, we removed the 10s condition from subsequent studies and will not present the 10s results. Delay conditions were assigned cyclically and logged by the server so that delay conditions were held constant for each user. From a series of pilot studies, we found a price point of $.04 per HIT was sufficient to generate adequate engagement.

After being assigned a delay condition, the worker would be presented with the first of five images for the selected HIT. In the background, the other four images would be downloaded at this time as well. The images themselves were approximately 25Kb each, which added negligible delays to the first image display time given average Internet connection speeds [2].

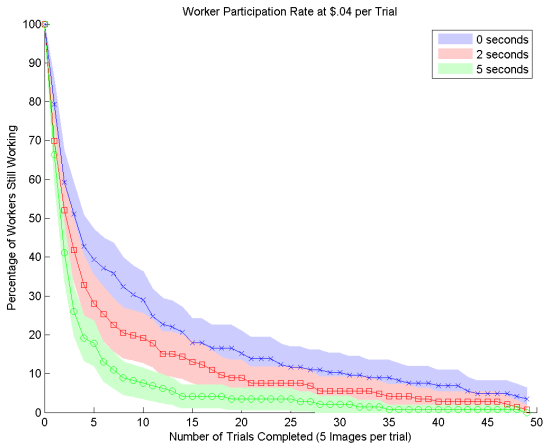

Figure 2. The survival graph for different download delays at the $.04/HIT price point. This graph shows the percentage of workers in each delay condition that completed various numbers of tasks (x-axis). The shaded areas designate the 95% confidence interval for each.

After completing the categorization and annotation of an image, workers would click the 'next' button (see Figure 1) and a JavaScript function would be invoked. That function would first clear the textbox and selected radio button, then, after the pre-assigned delay, the next image would be displayed.

At the end of each HIT, workers who have not yet completed 50 HITs have the option of completing additional HITs. This choice is made amidst a myriad of contextual factors that we have no way of measuring. However, by sampling a large number of workers (in this case, 583 unique workers chose to work on at least one HIT, completing a total of 3115 HITs), we can establish baseline models of workers' attrition rate. Figure 2 presents this attrition as a survival function calculated over the population of workers.

At the left edge of the survival graph, all three conditions are necessarily at 100%, since the data is restricted to workers who selected one of our HITs. The next point represents the number of workers who completed at least one HIT. The effects of the delay conditions arise quickly as the attrition curves diverge. As one would expect, increased delays lead to more workers opting not to continue completing our HITs. Without delays, 29% of workers completed at least 10 HITs. With a 2s delay, that

number drops to 19%, while only 7.5% of workers exposed to a 5s delay between every image will complete as many as 10 HITs. Were our image categorization and annotation tasks a real problem that needed crowd sourced responses, these completion rates would directly impact the consistency of the results and the time necessary to complete them.

The attrition curves can also be used to estimate the number of tasks that any given user should be expected to complete for a set of conditions. Since workers in our study were restricted to at most 50 HITs, we fit the completion counts for each delay to a censored Weibull distribution to account for this artificial ceiling. After fitting the data, we estimated the expected number of HITs a user would complete as the expected value of the model. By this method, we estimate that a user experiencing no additional delay would be expected to complete 11.3 tasks. Bandwidth limitations that led to 2s additonal delay per image would reduce the expected HIT completion rate to 8.4 per user and 5s delays would lead to only 4.7 tasks expected to be completed.

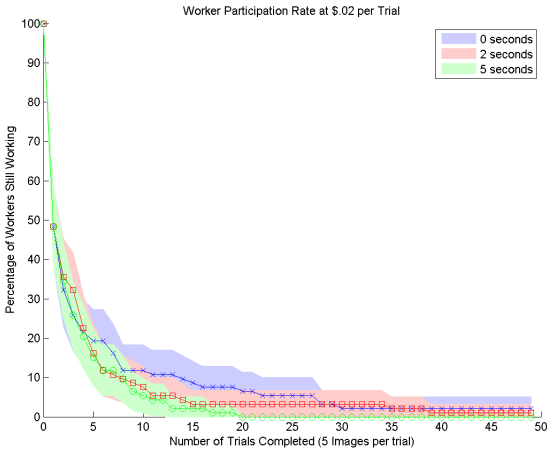

Figure 3. The survival graph for different download delays at the $.02/HIT price point.

Study 2: Cost Functions

To this point, we have generally ignored one of the most salient features of crowd sourcing, the pay rate. Having established a baseline survival function in study 1, we repeated the study twice more at different price points. This allowed us to examine how differing motivation, in the form of increased or reduced monetary payment, impacted workers' tolerance for delays. In the case of our image categorization task, this provides a direct measure of how worker payments impact engagement. In a real-world setting, this would allow us to calculate whether paying for additional bandwidth to reduce download delays would be more cost effective than increasing the task payment so that workers would put up with the longer delays.

One unsurprising effect of reducing the payment price for completed HITs is a drastic reduction in the number of workers who will engage with a task at all. Compared to the 583 workers who accepted one of our HITs when payment was set to $.04, only 372 workers accepted a HIT at $.02 (completing a total of 1095 HITs) during the study period.

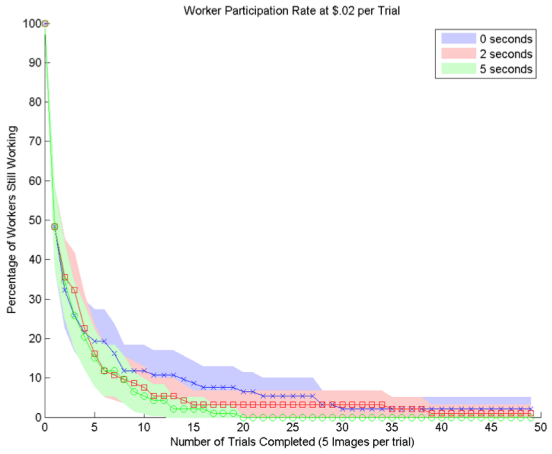

Figure 4. The survival graph for different download delays at the $.06/HIT price point.

As can be seen in Figure 3, the survival rate is lower in the $.02 trial than the $.04 trial for both 0s and 2s conditions. Interestingly, though, the rate of completion (see Figure 5) for the 5s condition is remarkably similar across the trials. One interpretation is that there is something of a survival rate floor driven by a small percentage of workers who value predictable tasks and will continue undeterred in the face of delays and low pay.

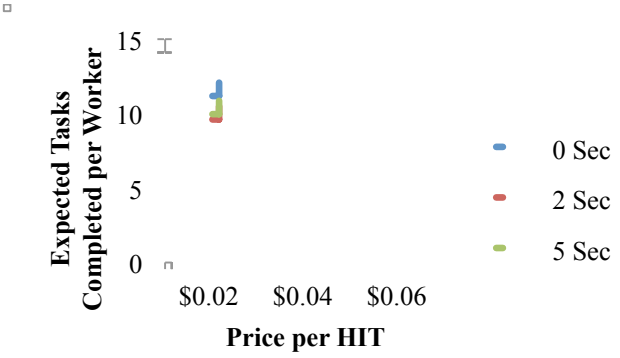

Figure 5. Cost curves for various completion rates given a fixed download delay. At $.04 per HIT, there is a much greater range in completion rates across latencies. At $.02 per HIT, workers complete very few tasks regardless of the delay whereas at $.06 per HIT any given worker can be expected to complete nearly 9 HITs regardless of the delay.

Increasing the HIT payment to $.06 resulted in 567 unique users completing a total of 4771 HITs. There was a notably higher survival rate for the 2 second and 5 second conditions, but has little noticeable effect on the 0 second case. In fact, at the $.06 price point, a log rank test indicated no significant difference in task completion rate across delay conditions. While it may seem surprising that a mere

$.02 increase in payment would completely eliminate the effect of our delay, keep in mind that this represents a 50% increase in payments received over numerous tasks.

Putting all three of these results together, we created a labor supply curve (see Figure 5) for our image recognition tasks. From this chart we can directly calculate the cost or savings in terms of task payments necessary to maintain a consistent level of user engagement across delays. We will return to these findings in our discussion section to better illustrate their utility.

Study 3: Latency Delay Costs

Having established a model of the cost of download delays, we turned our attention to the effects of network latencies. Researchers developing remote access applications [1, 23, 34], or exploring ways to offload computation to servers for mobile-cloud applications are particularly interested in how latencies effect user engagement. From a more applied perspective, developers debating about operating additional regional servers to cut down on latency, would certainly benefit from knowing the value of reduced latency on user engagement that these servers would provide.

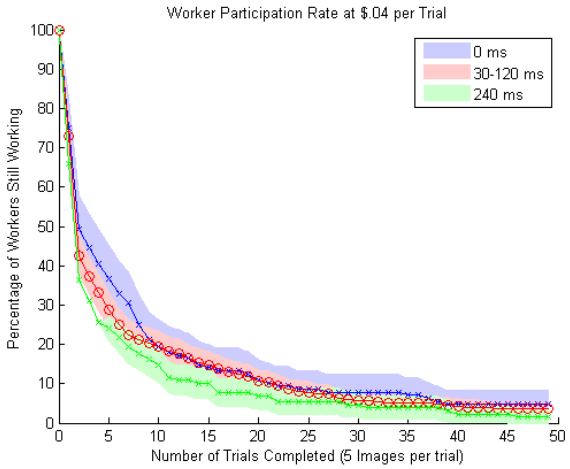

Figure 6. The survival graph for different network latency delays at $.04 per HIT.

Whereas Studies 1 and 2 introduced delays when switching between images in the image categorization tasks, the delays in this study were designed to imitate network latency as experienced during remote access computing. In remote access computing (e.g. Virtual Network Computing [27]) mouse and keyboard inputs are sent from the user's machine to a remote server where application processing occurs and the updated screen image is returned to the user's machine. The result is a lag between input and output that is determined by the time it takes to send information from the user machine to the server. For typical wired internet connections these delays range from tens to hundreds of milliseconds depending on the relative locations of the user and server [12, 20].

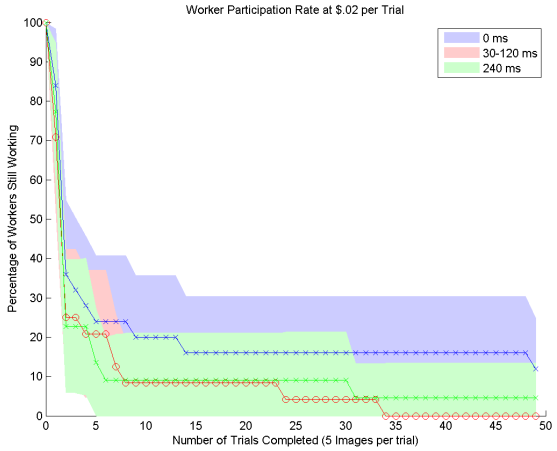

Figure 7. The survival graph for different network latency delays at $.02 per HIT.

To simulate remote computing lag, we introduced lag in the user's mouse motion. In particular, we wrote a JavaScript function that hid the worker's cursor, but logged its location and displayed a cursor image where the cursor had been a fixed delay time before. For our first study using latency delay, we paid $.04 per HIT and assigned workers to one of five latencies: 0, 30, 60, 120 or 240ms. In order to make the study as comparable to our original $.04 per HIT download delay study (see the Study 1 section), we made the series of HITs available until we had a comparable number of workers engaged in each condition (711 unique workers across 5 conditions, or nearly 142 per condition compared to 583 workers across 4 conditions for nearly 146 per condition in Study 1). The survival graph can be seen in Figure 6.

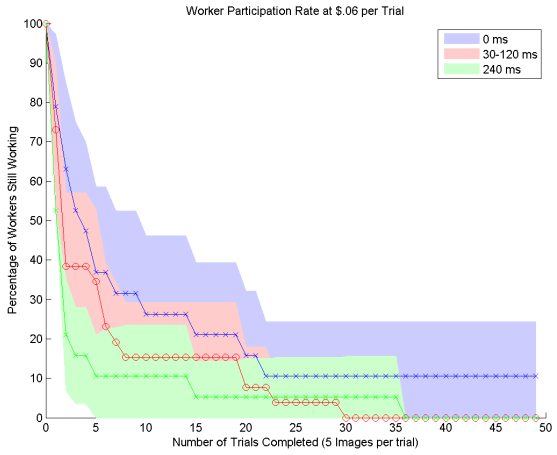

Figure 8. The survival graph for different network latency delays at $.06 per HIT.

After examining the results, we did not observe a significant difference between adjacent delays (i.e., log rank tests on 0ms and 30ms counts did not present significantly differences). Thus, for subsequent series of HITs, priced at $.02 and $.06 per HIT and shown in Figures 7 and 8,

respectively, only three delay conditions (0, 60, and 240ms) were run. For the sake of simplicity, we have left out the other delay conditions (30ms and 120ms) from the analysis and plots (Figure 6) of the $.04 condition as well.

For both the $.02 and $.06 condition trials, we were primarily interested in seeing how the expected task completion rates compared with the $.04 trial. For this reason, we ran the trials over a smaller sampling of workers. Additional HITs could always be added to increase our sample size, but after 253 and 238 unique workers (at $.02 and $.06, respectively) across three delay conditions, a Tobit regression algorithm modeling task completion rates indicated that payment price was a significant variable ($p < .0175$) for our latency delay trials.

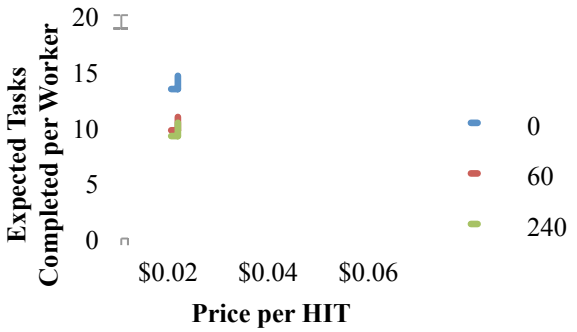

Figure 9. Cost curves for various completion rates given a fixed network latency delay. The curves here indicated a much more gradual variation across delays compared with the download delay case (see Figure 5).

The cost curves shown in Figure 9 paint a much more straightforward interaction between HIT payments and latency effects. Unlike the download delay curves (see Figure 5), the payment price does not dominate the expected completion rate. Rather, we see a gradual increase in engagement across price that is closely matched by increased engagement achieved by reducing delays. For example workers offered $.02 to complete HITs with 60ms delay would be expected to complete 5.9 tasks. Offering $.04 instead, would boost the expected task completions to 9.6. Alternatively, offering $.02 per HIT and reducing the latency to 0ms would similarly boost the expected completions to 8. 5.

DISCUSSION

In designing our study, we had two primary goals in mind. First, we wanted to compare the relative effect of download delays to network latency delays. The cost curves generated in the previous section (see Figures 5 and 9) highlight the disparate effects that different delay types have on user engagement. For the relatively simple and low paying image classification tasks that we used, it is clear that download delays of a few seconds result in a much stronger effect than up to a quarter second of network latency. Thus, placing a higher emphasis on maintaining quick downloads for this type task is more important than achieving minimal lag. While these results may not hold for other applications, a developer could follow the methodology presented here to reach conclusions about the relative impact of these delays on other tasks.

Our second goal was to demonstrate how these results could be used to optimize system design choices. We will explore that process in the following section.

Latency Cost Analysis

One of the most obvious approaches to reducing network latency is to operate additional servers located in physically disparate regions. However, running additional servers comes at a cost. Here we will use the results from our study to demonstrate how our methodology could be used to better inform decisions about when operating additional servers would make financial sense.

To estimate the costs and latencies of operating additional servers in various regions, we have chosen to use Amazon Web Service (AWS) as a representative service platform. The costs presented in Table 1 are calculated using AWS Total Cost of Ownership Calculator [3]. The latencies listed were measured from a city in the Eastern half of the United States using the CloudPing website [7]. While these measurements may not be perfectly representative of the latency to an AWS server, they are sufficient to demonstrate the cost analysis.

Given a particular set of application requirements, these measurements and costs combined with results like those presented in our studies can be used to make informed choices about server provisioning.

Region	Latency	Cost
US-East (Virginia)	49 ms	$178
US-West (California)	93 ms	$245
US-West (Oregon)	94 ms	$178
Europe (Ireland)	120 ms	$195
Europe (Frankfurt)	129 ms	$210
Asia Pacific (Singapore)	279 ms	$341
Asia Pacific (Sydney)	232 ms	$341
Asia Pacific (Japan)	186 ms	$289
South America (Brazil)	149 ms	$351

Table 1. The Amazon AWS regions with estimated network latencies and the 3-year cost [3] of a VM server in that region. Latencies are measured from an Eastern US city [7].

As an example, let us imagine a scenario in which a company wants to crowd source the translation of German text to English. The company plans to crowd source workers in the US and Germany for the task. They are on a tight deadline and want to engage as many workers as

possible in the next 24 hours. They also are working on a tight budget and cannot afford to spend extravagantly.

For simplicity's sake, we'll assume that crowd sourced workers the company is targeting in both the US and Germany behave similarly to the Mechanical Turk workers in our study and that the text translation task is of similar complexity to the image classification task described here.

Two conceivable options are (1) to engage a single Amazon Web Service environment in the US-East region or (2) to operate two servers, one in the US-East region and one in the Europe (Frankfurt) region. Looking to Table 1, the first setup will cost $178, while the two-server setup will cost $388. The single server setup will impose an 80ms (129ms-49ms) extra lag on the German workers that will be absent in the second option. Comparing this increased lag to our test conditions in Study 3, it is closest to the 60ms case. For simplicity's sake, we will continue our analysis using 60ms as the condition of interest. Table 2 shows the expected task completion rates at different payment prices for these delays.

			Expected Completion Rate	
Server	Cost	Lag	$.02 / HIT	$.06 / HIT
US-East	$178	60ms	5.9	10.9
US-East & Frankfurt	$388	0ms	8.5	14.6

Table 2. The costs of different server configurations. The 'Cost' and 'Lag' columns are the prices and delays associated with different server configurations. The 'Expected Completion Rate' columns show the number of HITs a user would be expected to complete at a give payment rate.

The assumptions we make about the available workers and number of tasks we need to complete, greatly affect this calculation. For example, if we consider the available supply of qualified workers to be limitless, then the lowest cost solution will always be a single server and minimal payment rate. The low, 5.9 expected completion rate, does not matter if there are enough available workers to engage. On the other hand, if we only seek to maximize the rate of task completions, then paying for a second server and maximizing the payment rate will always be the optimal course of action. The higher cost will result in an expected 14.6 tasks per worker. The more interesting cases, however, lie between these extremes.

If we consider that there will be a finite number of qualified workers available any given day, our calculations change. In our study, we engaged roughly 500 workers per day in our study. Table 3 shows the number of HITS that we would expect to have completed given the different server configurations and payment amounts. In this case, if completing 5000 tasks each day is required, the most efficient option would be to employ a single server with a 60ms lag and to pay a $.06 rate per HIT.

	$.02 / HIT	$.06 / HIT
60ms	2950	5450
0ms	4250	7300

Table 3. The expected number of HITs that would be completed by a population of 500 workers, given different server configurations and payment prices.

If slightly more workers are available, the tradeoff point at which engaging a second server becomes economical can also be calculated from these estimates. The cost of the first

$178 + $.06 * N = $388 + $.02 N, for N = 5250

Thus, if more that 5250 tasks need to be completed, if would be more economical to engage a second server and pay a lower per HIT rate. If few than 5250 tasks need to be complete, paying a higher rate per HIT outweighs the benefit of reducing the lag.

FUTURE WORK

Having just demonstrated a methodology for modeling user engagement in response to varying types of system response delays, an obvious next step would be to more rigorously validate the process. Given that most of our series of HITs were posted over the course of a single day, our results are certainly susceptible to day to day fluctuations of the Mechanical Turk market. A particular glut of other high paying tasks could have biased the response of workers while our HITs were available. Only by repeating the process several times can we gain additional confidence in the validity of our findings.

Similarly, it would be useful to explore other types of tasks with our methodology. While image categorization tasks are canonical examples of crowd sourcing tasks that made for a straightforward design, it would be interesting to explore longer, more complicated applications. The Mechanical Turk platform allows access to externally hosted sites (which we used) so there is no reason that more complex applications could not be examined. Undoubtedly, additional challenges would arise in having crowd workers engage in less familiar activities.

Finally, we would like to validate our approach using actual network delays rather than approximations of them generated by JavaScript functions in the worker's browser. Again, by making use of externally hosted servers, we should be able to direct the HIT traffic through controlled networks. For this first exploration into using crowd sourcing to explore engagement we opted not to employ a network emulator as it seemed to be an additional source of complexity.

CONCLUSION

In this paper we have demonstrated that crowd sourcing platforms can be used to measure the user engagement effects of various sources of system response delays. Whereas previous efforts to measure the effect of system

response delays have focused on performance or feelings of satisfaction, which are in turn used as proxies to estimate how user engagement will be affected, we have effectively bridged that gap. By following the methodology we have presented, developers can get a true sense of whether users will choose to interact with their system based on a fixed set of network conditions.

Beyond that, we have demonstrated how the results from our methodology can be used to inform complicated choices that revolve around financial investments and user engagement. While we are fully aware that our estimations are by no means unassailable, we are unaware of other approaches that offer as much promise to developers who are faced with difficult decisions about system deployment.

ACKNOWLEDGMENTS

This work was supported by the National Science Foundation under grant IIS-1065336 and the NSF Quality of Life Technology Engineering Research Center under grant EEC-0540865. Any opinions, findings, conclusions or recommendations expressed in this material are those of the authors and should not be attributed to Carnegie Mellon University or the funding sources.

REFERENCES

1. Yoshihisa Abe, Roxana Geambasu, Kaustubh Joshi, H. Andres Lagar-Cavilla, and Mahadev Satyanarayanan. 2013. vTube: efficient streaming of virtual appliances over last-mile networks. In *Proceedings of the 4th Annual Symposium on Cloud Computing* (SOCC '13). http://dx.doi.org/10.1145/2523616.2523636

2. Akamai. 2015. State of the Internet, Second Quarter. 8, 2 (2015). Retreived August 11, 2015 from https://www.stateoftheinternet.com/downloads/ pdfs/2015- q2- state- of- the- internet- report.pdf

3. Amazon. 2015. AWS Total Cost of Ownership (TCO) Calculator. Retrieved September 10, 2015 from https://aws.amazon.com/tco-caclulator

4. Rajesh Krishna Balan, Darren Gergle, Mahadev Satyanarayanan, and James Herbsleb. 2007. Simplifying cyber foraging for mobile devices. In *Proceedings of the 5th international conference on Mobile systems, applications, and services* (MobiSys '07), 272-285. http://dx.doi.org/10.1145/1247660.1247692

5. Raymond E. Barber and Henry C. Lucas, Jr.. 1983. System response time, operator productivity and job satisfaction. *Commun. ACM* 26, 11 (November 1983), 972-986. http://dx.doi.org/10.1145/182.358464

6. T. W. Butler. 1983. Computer response time and user performance. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems* (CHI '83), 58–62. http://dx.doi.org/10.1145/800045.801581

7. CloudPing. 2012. CloudPing. Retrieved September 10, 2015 from http://www.cloudping.info

8. Gary L. Dannenbring. 1983. The effect of computer response time on user performance and satisfaction: A preliminary investigation. *Behavior Research Methods & Instrumentation* 15, 2 (March 1983), 213–216. http://dx.doi.org/10.3758/BF03203551

9. Mitchell Grossberg, Raymond A. Wiesen, and Douwe B. Yntema. 1976. An experiment on problem solving with delayed computer responses. In *IEEE Transactions on Systems, Man and Cybernetics,* Vol. SMC-6. 219–222. http://dx.doi.org/10.1109/TSMC.1976.5409241

10. Jan L. Guynes. 1988. Impact of system response time on state anxiety. *Commun. ACM* 31, 3 (1988), 342–347. http://dx.doi.org/10.1145/42392.42402

11. John A. Hoxmeier and C DiCesare. 2000. System response time and user satisfaction: an experimental study of browser-based applications. In *Proceedings of the Association of Information Systems Americas Conference* 2, 1–26. http://dx.doi.org/10.1.1.99.2770

12. Junxian Huang, Feng Quian, Alexandre Gerber, Z Morley Mao, Subhabrata Sen, and Oliver Spatscheck. 2012. A close examination of performance and power characteristics of 4G LTE networks. In *Proceedings of the 10th international conference on Mobile systems, applications, and services* (MobiSys '12), 225–238. http://dx.doi.org/10.1145/2307636.2307658

13. Panos Ipeirotis. 2010. Demographics of Mechanical Turk. *New York University Working Paper*.

14. Julie A. Jacko, Andrew Sears, and Michael S. Borella. 2000. The effect of network delay and media on user perceptions of web resources. *Behavior & Information Technology* 19, 6: 427–439. http://dx.doi.org/10.1080/014492900750052688

15. Aniket Kittur, Ed H. Chi, and Bongwon Suh. 2008. Crowdsourcing user studies with Mechanical Turk. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (CHI '08), 453-456. http://dx.doi.org/10.1145/1357054.1357127

16. Werner Kuhmann. 1989. Experimental investigation of stress-inducing properties of system response times. *Ergonomics* 32, 3 (March 1989), 271–280. http://dx.doi.org/10.1080/00140138908966087

17. John P. Klein and Melvin L. Moeschberger. 2003. *Survival Analysis: Techniques for Censored and Truncated Data, Second Edition.* Springer, New York, NY, USA.

18. Albert Lai and Jason Nieh. 2006. On the performance of wide-area thin-client computing. *ACM Trans of Com Sys.* 175-209.

19. Walter S. Lasecki, Jeffery M. Rzeszotarski, Adam Marcus, and Jeffrey P. Bigham. 2015. The effects of sequence and delay on crowd work. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (CHI '15), 1375-1378. http://dx.doi.org/10.1145/2702123.2702594

20. Ang Li, Xiaowei Yang, Srikanth Kandula, and Ming Zhang. 2010. CloudCmp: comparing public cloud providers. In *Proceedings of the 10th annual conference on Internet measurement* (IMC '10)*,* 1–14. http://dx.doi.org/10.1145/1879141.1879143

21. Robert B. Miller. 1968. Response time in man-computer conversational transactions. In *Proceedings of the December 9-11, 1968, fall joint computer conference, part I* (AFIPS '68*),* 267-277. http://dx.doi.org/10.1145/1476589.1476628

22. Brad A. Myers. 1985. The importance of percent-done indicators for computer-human interfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (CHI '85), 11-17. http://dx.doi.org/10.1145/317456.317459

23. Fiona Fui-Hoon Nah. 2004. A study on tolerable waiting time: how long are web users willing to wait? *Behavior & Information Technology* 23, 3: 153-167. http://dx.doi.org/10.1080/01449290410001669914

24. Jakob Nielsen. 1993. *Usability Engineering.* Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

25. Gabriele Paolacci, Jesse Chandler, and Pg Ipeirotis. 2010. Running experiments on amazon mechanical turk. *Judgment and Decision Making* 5, 5: 411–419. http://dx.doi.org/10.2139/ssrn.1626226

26. Judith Ramsay, Alessandro Barbesi, and Jenny Preece. 1998. A psychological investigation of long retrieval times on the World Wide Web. *Interacting with Computers* 10, 1: 77–86. http://dx.doi.org/10.1016/S0953- 5438(97)00019- 2

27. Tristan Richardson, Quentin Stafford-Fraser, Kenneth R. Wood, and Andy Hopper. 1998. Virtual network computing. *IEEE Internet Computing* 2, 1 (1998), 33–38. http://dx.doi.org/10.1109/4236.656066

28. Joel Ross, Lilly Irani, M. Six Silberman, Andrew Zaldivar, and Bill Tomlinson. 2010. Who are the crowdworkers?: shifting demographics in mechanical turk. In *CHI '10 Extended Abstracts on Human Factors in Computing Systems* (CHI EA '10), 2863-2872. http://dx.doi.org/10.1145/1753846.1753873

29. Avi Rushinek and Sara F. Rushinek. 1986. What makes users happy? *Commun. ACM* 29, 7: 594-598. *http://dx.doi.org/10.1145/6138.6140*

30. Mehadev Satyanarayanan, Paramvir Bahl, Ramon Caceres, and Nigel Davies. 2009. The case for VM-based cloudlets in mobile computing. *Pervasive Computing* 8, 4 (2009), 14–23. http://dx.doi.org/10.1109/MPRV.2009.82

31. Ben Shneiderman. 2005. *Designing the User Interface: Strategies for Effective Human-Computer Interaction.* Vol. 3. Addison-Wesley.

32. Steven L. Teal and Alexander I. Rudnicky. 1992. A performance model of system delay and user strategy selection. In *Proceedings of the SIGCHI conference on Human factors in computing systems* (CHI '92), 295–305. http://dx.doi.org/10.1145/142750.142818

33. M. Thum, W. Boucsein, W. Kuhmann, and W. J. Ray. 1995. Standardized task strain and system response times in human-computer interaction. *Ergonomics* 38, 7: 1342–1351. http://dx.doi.org/10.1080/00140139508925192

34. Niraj Tolia, David G. Andersen, and Mahadev Satyanarayanan. 2006. Quantifying interactive user experience on thin clients. *Computer* 39, 3: 46–52. http://dx.doi.org/10.1109/MC.2006.101

35. Michael Toomim, Travis Kriplean, Claus Portner, and James Landay. 2011. Utility of human-computer interactions: toward a science of preference measurement. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (CHI '11), 2275–2284. http://dx.doi.org/10.1145/1978942.1979277