Comparing Web Interaction Models in Developing Regions

Jay Chen
New York University
jchen@cs.nyu.edu

Aditya Dhananjay
New York University
aditya@cs.nyu.edu

Saleema Amershi
University of Washington
samershi@cs.washington.edu

Lakshmi Subramanian
New York University
lakshmi@cs.nyu.edu

ABSTRACT

Internet connections in developing regions are scarce and often unreliable. While options for connecting to the Internet are gradually being realized, progress is slow. We observed people performing web search and browsing in a low bandwidth environment in Kerala, India. We found that people in this environment experienced frustration and boredom while waiting for page loads compared to typical experiences in the developed world. Following these observations, we conducted a formal study with 20 participants at the same location comparing the conventional web search and browsing process with an asynchronous queueing model. Participants using the asynchronous queueing system performed as well as the status quo in terms of the number of tasks completed, and we observed greater interaction and information viewed for the asynchronous system. Our participants also preferred the asynchronous system over conventional search. Finally, we found evidence that the asynchronous system would have greater benefits in environments where the network is even more constrained.

1. INTRODUCTION

Slow, expensive, or non-existent Internet access is a fact of life for millions of people in the developing world where the physical infrastructure has yet to catch up to the increasing demand. In these regions connectivity is often intermittent (disconnected for a long time over large timescales or a short time over short timescales) due to power cuts and many other issues [5, 36, 3]. Bandwidth is generally an expensive and rare commodity because none of the conventional wireline connectivity solutions (fiber, broadband and dial-up) are economically viable for rural regions with low user densities [25]. The recent emergence of new low-cost connectivity solutions using long-range wireless technologies (cellular, WiMax [39], long-distance WiFi [28]), and delay tolerant mechanical backhaul networks (connectivity via busses, motorbikes, etc. driving in a loop) [44, 3] provide hope for rural connectivity. However, even after connectivity to the Internet is established, the connection is most often still slow as a result of either low bandwidth, high latency, intermittency or all of the above.

Slow connections mean long delays for page loads (on the order of tens of seconds or minutes for low bandwidth or high latency links), and a frustrating experience. The impact of such slow connections is relatively mild for asynchronous applications such as e-mail or downloading large files, but impractical for interactive applications such as web browsing.

Existing work on bringing the web to developing regions has generally focused on addressing either system level infrastructure issues or specialized user interfaces (e.g. One-handed thumb use on small devices) [18, 21, 20]. However, Research that focuses on constrained web search and web access is either dated [23, 26] or, in the case of more recent work, is in the context of mobile devices [7, 41, 34, 43, 18, 21]. While there is extensive literature about web usage in general, particularly in the WWW community [1], until very recently little effort has been directed at understanding the unique characteristics of Internet use in developing regions [13, 9, 40]. A few network constrained applications have been designed and implemented by splitting the application into a fully synchronous front end and an asynchronous back end [40, 24, 37, 8]. However, the effects of applying this asynchronous model of interaction and common web optimization techniques on users are unknown.

In this paper, we use the term web *interactions* as a shorthand for the combination of web search and browsing activities that people engage in while using the web. The contributions of this work are to understand the issues people face during their web interactions in developing regions particularly as a result of slow connections, and compare two different web interaction models: the conventional web model and an asynchronous model.

In this paper we begin by surveying the related work. We then describe our formative study where we observe the performance and behavior of people using the conventional web interaction model. After an overview of our asynchronous web search interface [8], we present results from a formal study comparing the the systems along with basic web optimization techniques. Finally, we discuss our results and how they may generalize to other constrained network environments.

2. RELATED WORK

There has been considerable work in web search in general, beginning with user search behavior to identify search prefer-

ACM DEV'10, December 17–18, 2010, London, United Kingdom.

ences within the "search and refine" framework [6]. As search systems and user interfaces evolved, studies have tracked user behavior changing along with technology trends [27, 38]. Characterizing traffic patterns of groups of users in developing regions has been explored in isolated cases [9, 32], and more recently attempts have been made to do so at larger scales [13]. However, relatively little is known about web *interaction* and *behavior* in developing regions particularly among the poorly connected [40]. We focus on exploring these behaviors and considering the effects of performance optimizations in this space.

In the context of developing regions, search and browsing has been largely ignored in favor of research devoted to establishing Internet connectivity in the first place [30, 10, 28]. Even this first step is difficult because economic constraints must be taken into account, meaning low cost solutions are more likely to scale [12]. Examples of these hardware solutions include long distance wifi [28] and low earth orbiting satellites. These systems allow connectivity, but in many cases the poor connection quality is unable to be masked by conventional means such as caching and compression. Software solutions solely at the application layer such as loband [22] (which provide compressed text-only page rendering) do not adequately address the developing countries' problem due to the lack of support for delay tolerant transport mediums. One of the earliest systems in this area has been TEK [37]. TEK allows queueing of pages for asynchronous download over simple mail transfer protocol (SMTP), and local search functionality.[1] However, it is unclear which mechanisms benefited users to what extent and how generalizable those results are to different environments.

More recently, the potential for information access via mobile devices has generated substantial interest in the development and web search communities as these devices mature into web search platforms [35, 17, 16, 42]. Due to the imbalance between the size of content on the web and the capabilities of mobile devices, some efforts have opted to modify the web content itself to be more mobile friendly [4]. Mobile web standards have been designed and refined to that end, but this approach places the burden upon content developers to create and maintain separate versions of their websites.

Alternative work in improving web access in the developing regions has looked into increasing the value of each piece of computing hardware by multiplexing these physical resources via software systems [29, 2]. These systems improve the efficiency of computer hardware at the user endpoint rather than the usage of bandwidth provided by networking hardware in the infrastructure. Still other efforts in this area are designed for specific network configurations such as peering behind individual upstream bottlenecks [33], collaborative caching behind a shared bottleneck link [15], or wireless mesh networks [14]. Unlike these works, we focus on exploring user behaviors and interaction with existing web interaction models and investigate the benefits of a different interaction model and optimizations.

3. CONVENTIONAL SEARCH

To understand the challenges with web search in under-provisioned settings under the conventional web search model, we observed university students and staff performing web search from behind a heavily shared bottleneck connection on their campus in Kerala, India. The network available on the campus was a broadband 8Mbps connection. However, this single connection was being shared across 400 machines by over 3000 students, staff, and faculty. Furthermore, only 750Kbps in total bandwidth was allocated to students and staff. A simple estimate of the worst-case average bandwidth available per machine is approximately 1.9Kbps. This is abysmally slow even compared to dial-up (56.6Kbps). The average-case bandwidth available across all users was hard to determine as it fluctuated depending on the number of concurrent users at the time and priority queueing effects at the gateway router. All traffic was routed through a web proxy with a 20GB disk cache. Our study participants are well educated. While this is not representative of all users in the developing world, we hypothesize that less educated users may have fewer coping mechanisms.

We used an existing machine in the university's computer lab running Windows XP and Internet Explorer 7. The machine and the network were powered by backup generator to avoid power outages. The experiments were performed between 12:00pm and 10:30pm. We did not artificially constrain the time of day as the fluctuations in bandwidth were themselves part of the phenomenon of interest.

Fifteen participants (11 male) between 19 and 25 years of age were observed in this study. All participants were enrolled in college, college graduates, or had completed their masters. Five participants were students and ten participants were staff. The participants' fields of study included computer science, electrical engineering, commerce, and business management. Every participant was self-reported to be at least moderately experienced with web search and was able to converse and browse the web in English. The Internet connection that the participants were familiar with varied between dialup and broadband.

Each participant was first given a simple demographic and search experience questionnaire. Questions included level of familiarity with the Internet, average Internet usage per day, and comfort while searching the web. Then the participant was asked to use the Internet for web search using the search engine of their choice to pursue any search topics of their choice. Participants were not informed beforehand as to the exact duration of time available so they would not feel rushed (but were given up to 15 minutes to search). Finally, participants were given a brief semi-structured interview (around 10 minutes) about their experience. Participants were observed, videotaped, and a screen-capture video of their screens was recorded.

3.1 Results

The actual bandwidth the participants experienced varied significantly from 20Kbps during peak hours in the early to mid afternoon to 200Kbps in the evening. The reason the bandwidth was higher than the worst-case is likely because not all machines were accessing the network simultaneously. Here we focus our analysis on the participants' web search behavior experienced under these conditions.

3.1.1 General Behaviors

Search engine results pages tended to load quickly (under five seconds) due to the mostly-text content and low latency

[1]TEK was designed for and studied with people who were economically deprived so the goals and results of this study are not identical.

to the search engine servers. None of the users had complaints about the search results page load times during the experiment. The requests for general pages (pages not provided by the search engine) took varying amounts of time to load (ranging from one to 240 seconds) depending on server latency, network congestion, and page contents. The overall statistics for idle time of all page loads in seconds are: mean 12.62, median 4, stddev 26.27. These numbers are inclusive of hits in the local web proxy's cache (10-25% hit rate) and any compression implemented by the accessed web servers. We observed that although many requests were satisfied quickly with 70% pages loading in under 10 seconds, a huge variance the load times of different pages exists with some pages taking up to five minutes to load. Also, we observed that quickly loading pages were mostly navigational or pages with very little information content.

Participants had no complaints with pages that took less than five seconds to load and were generally only affected if page loading time exceeded ten seconds. When this happened, participants were observed sighing, staring at a blank page loading screen, leaning back, and trying to make conversation with the observer while waiting. This behavior along with self-reported boredom and frustration (e.g. 'I feel angry') generally increased as the page loading time increased. When page load times were over one minute, several participants reported that under normal circumstances they would do something else, give up, wait to search at a later time, or use a faster connection elsewhere. Four participants reported that they had access to a home broadband connection, and all had access to and were aware of a broadband pay-per-use PC cafe on the campus. As found in previous research [6], participants generally preferred looking only at the first page of results, electing to modify their search query rather than go on to a second page. None of the participants used any advanced search engine options, though most were aware of their existence. Two participants, when asked, revealed that advanced search was not worth the effort to use, preferring instead to iteratively modify their query. Four participants performed image search during the experiment, and two others mentioned they often perform image search.

3.1.2 How Did People Seek to Alleviate Wait Times?

Seven participants (47%) opened multiple windows (the web browser we installed at the school did not have tabbed browsing), and two more reported that they occasionally would. Six reported that outside of the experiment they commonly multi-tasked with other activities including listening to music, using offline applications, reading, or talking to a friend. Some participants who opened multiple windows switched between them while waiting for pages to load.

3.1.3 How Successful were People at Saving Time?

One of the experimenters coded our video data to approximate participants' total idle and busy periods. A participant was defined as busy when he was observed reading the contents of the page or performing any navigational mouse action, and defined as idle otherwise. We found that people who opened multiple windows and switched between them wasted less time (i.e., spent an average of 74% less time being idle). Participants who opened multiple windows, but did not switch between them performed slightly better than those who used a single window (28% less idle time on average). Only when the page loading rate was faster than the user's rate of information consumption (reading speed) was the page load time disguised. We found that some of the improved performance exhibited by the opening multiple windows is an artifact of our particular experimental environment. In our environment, connections are throttled across a shared pool of bandwidth. As a result, when people open multiple windows they effectively increased the amount of bandwidth available to them. What is interesting is that by multi-tasking users self-impose asynchrony to the web search process. Also, even if the bandwidth was not shared across users, people would see gains because data would continue to download on other pages while they view pages already available.

3.1.4 Benefits of Caching and Compression

During our study, the web proxy only had a cache size of 20GB, and a hit rate of 10-25%. Approximately 27.5% of web servers on the Internet compress files they serve [31]. To estimate the best-case scenario for compression we compressed all files downloaded by our participants during the experiment using a simple compression algorithm (gzip). Unsurprisingly, we found that compressing benefits text files the most (up to 80%) compared to image files which are already compressed, but text files only represent a small (and diminishing) proportion of total webpage sizes on the web [19]. From our idle time measurements and interviews, the existing caching and compression were unable to mask the network latency for our participants. Even with state of the art caching and compression the required orders of magnitude improvement needed to shield the user from delays would not be achieved for intermittent connections.

3.1.5 Observations of Well-known Results

First, most of the searches our participants performed were for textual information, thus most images were not useful despite taking up the majority of download times. This supports previous work that a text-only browsing option could improve satisfaction if images are unwanted [37, 22]. Second, web browsers by default currently only have a progress bar to indicate page loading progress. This feature was not helpful to our participants because it presented only a vague estimate of the time to complete a page load. One participant even claimed to estimate the time to load a page manually based on the rate at which the progress bar filled up. Providing people with a more accurate page load time estimate could allow them to multi-task more efficiently.

We also noticed that our participants often had problems entering effective search queries on their first attempt; their search sessions often required several query reformulations. This affected our participants only slightly, but with high latency or intermittent connections this would be more of an issue. Finally, assistance for search query construction (ie. Google Suggestions) was not always immediately available due to network sluggishness, and when suggestions did appear, they were never explicitly used. When asked why they did not use query suggestions, participants responded that they preferred to iteratively refine their search terms based on previous results.

4. ASYNCHRONOUS QUEUING MODEL

In this section, we describe an asynchronous queueing model for web search and browsing. We discuss this model in the context of RuralCafe [8], a system designed to sup-

port asynchronous queueing. The core feature we are studying (asynchronous queueing) is independent of this particular implementation (i.e. we could have implemented local search and prefetching on top of TEK instead).

RuralCafe uses a simple proxy architecture to provide asynchrony; end-hosts in a local area network connect to the Internet over the poor connection using proxies. One proxy is placed at either end of the connection, and all traffic to and from the end-hosts traverse the local proxy before being tunneled across the link to the remote proxy. The remote proxy connects to the Internet using a direct network connection. The local proxy is equipped with a large local store that the client can locally search without using the network. When user directed requests require the network they are queued and dispatched to the remote proxy. The remote proxy continually prefetches pages for return to the local proxy in the background.

The interface of our web search system consists of three components presented to the user in the form of browser frames: *Search Frame*, *Request Queue Frame*, and *Active Frame*. The user interacts with each of these frames as illustrated in Figure 1.

4.1 Local Search and Query Refinement

Local search assists the user in offline query refinement by allowing a person to search through the cache at the local proxy. In addition, offline query construction assistance is provided in the form of suggested queries similar to those of major search engines. Currently, suggested queries are returned by the local proxy (using a simple term frequency database) which also does not require any network resources.

The user is presented with a simple *Search* frame to perform local searches and query refinement. We made the design decision to dedicate screen real estate to the search frame to encourage the use of local search which we believed would improve the usability of the feature. Requests made in this frame may either be served locally without use of the network or added to the queue in the *Request Queue* frame. Once a local search is performed, the local proxy returns a list of links to pages in the cache along with suggested queries (Suggested Queries & Local Results in Figure 1). The user may click on the links to view the pages or click the suggestions for another local search. The user could also queue the query for download if the local results were unsatisfactory.

4.2 Queueing

The main difference in the asynchronous model is the use of queueing to decouple user requests with network availability. What this means is that for pages that are not in the local proxy's cache (require network access), the user can queue up the page for download and return to it after it is downloaded. While the server processes the queue, a person can continue to perform other searches or queue up more page requests. All requests that require network access must be added to this queue.

The *Request Queue* frame in Figure 1 displays the list of pages queued by the user and list of the queued items and their status for browsing. If a page is being downloaded or waiting in the queue, its associated expected completion time is displayed. If a page in the queue is completed, a link is displayed to access the page. If the link is clicked, the results are served locally and presented in the *Active* frame.

A person is informed of the pages that must be added to the queue and then is free to do so if the page is desired. A person is free to add or remove pages from their queue at any time. We dedicated space in our user interface to display the queue so users could always see the status of their requests and act upon the information easily.

4.3 Remote Requests and Prefetching

The *Search* frame also allows users to queue up remote requests. The interface is a search box and two buttons for queueing either a search or a URL to be downloaded. The user can also select the types of pages to download either text only or text and images, and the prefetching depth if prefetching is enabled. The default is for text only and prefetching depth of one page.

Each request is associated with a *quota* that represents the maximum number of bytes that is allocated as the overall response budget for a query. Each request also has a limit to how many link levels deep to prefetch. These limits are implicitly controlled by the user with the radio buttons below the search box. With these two limits, the remote proxy prefetches a response to a page request recursively until the quota is met or the depth limit is reached. Once complete, all downloaded pages are returned to the local proxy to be incorporated into its cache.

5. ASYNCHRONOUS SEARCH

To study the effectiveness of the various modifications on improving the search experience in high latency, low bandwidth environments, we conducted a within-subject experiment comparing two versions of our system to search using a conventional web browser (Firefox 3.0) with text-only enabled by default which we refer to as *Conventional*. We use RuralCafe with Firefox for both asynchronous systems. The first asynchronous queuing system had only the local search capability enabled (*LocalSearch*) because we believed that local search alone may provide performance benefits to users. This system was the most closely related to TEK [37] in terms of core features (queueing and local search). The second asynchronous queueing system had both local search and prefetching enabled (*Prefetch*).

Twenty students (6 male) between 22 and 24 years of age from the same university as our preliminary study were recruited as volunteers from the student body to participate in this study. Our participants could not uniformly commit more than one and a half hours to the study because of classes and scheduling. Due to these time constraints, we formed two gender balanced groups of ten participants each wherein each participant used one system (*LocalSearch* or *Prefetch*) for 30 minutes and *Conventional* for 30 minutes. The order of the conditions was counterbalanced in each group so as to reduce ordering effects.

Participants were first given a short demographic questionnaire before the start of the study.

During each condition, participants were given one of two sets of five informational search tasks each. Each task was designed to require multiple searches and each task set had goals of comparable difficulty resembling both focused and open ended search tasks such as:

"What state in India has the largest population, and what is the population?"

and

"Pretend you are trying to find a cheap digital camera,

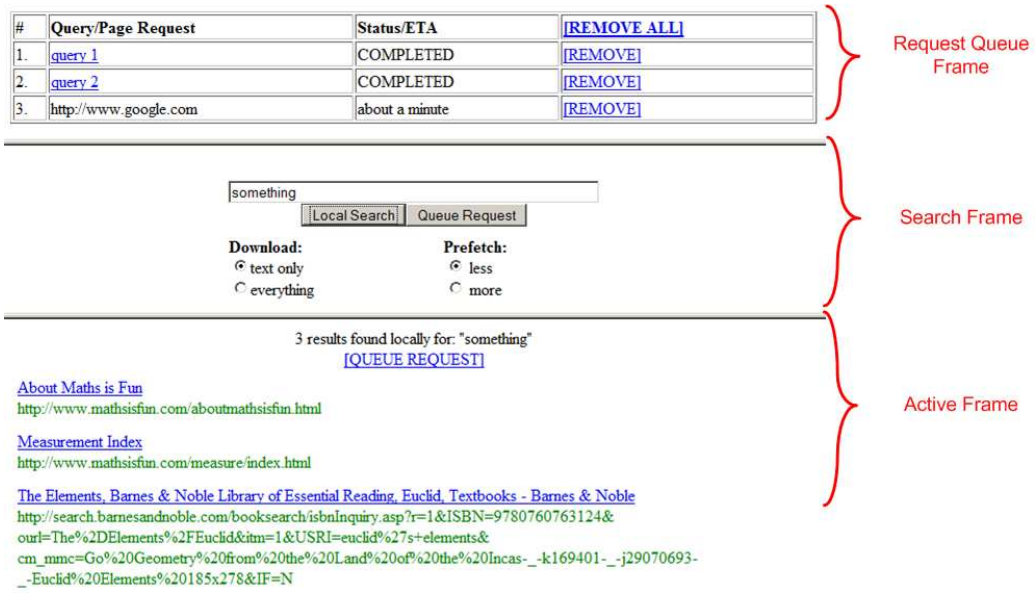

Figure 1: User interface. The three frames are labeled in red. The user is free to interact with any of the frames at any time.

find two possible cameras along with their prices."

The five tasks were given to each participant all at once before starting each condition and participants were told to complete as many tasks as possible. Task completion was self judged by the participant, the answer written down, and confirmed by the observer. Giving participants five tasks at once was intended to determine how well our mechanisms helped with multitasking.

A final exit questionnaire was given at the end of the study asking questions including their preference for the modified or the standard web browser and the three best and three worst things about the modified system they used. All participant actions on the web browser were logged during the study.

The only difference in the environment from our initial formative study was that the bandwidth for each experiment was throttled to 50Kbps for consistency. This environment represents a low bandwidth connection as opposed to a high latency or intermittent connection. The browser used for this study was Firefox version 3.0 with multiple tabs enabled. Also, the screen available for use in this study were considerably smaller (15 inches) than in the formative study (19 inches).

We did not have access to the contents of the existing cache, so to provide a realistic cache, we manually warmed a local proxy cache by iterating once through the task sets in our study. For our study we bypassed the existing proxy completely, and used our warmed cache in its place. We used a fresh copy of this warmed cache for each participant for each condition *including the baseline* for fairness. From our results the cache hit rate was approximately 19.5%, which is comparable to the actual cache hit rate of the Squid proxy cache at the university (10-25%).

5.1 Quantitative Results

We perform all of our analyses on the logged data obtained from our study using mixed-model analyses of variance with repeated measures because our experiment was a mixed between- and within-subjects factorial design (with participants in group one using *LocalSearch* and *Conventional* and participants in group two using *Prefetch* and *Conventional*). All of our models include Method (*LocalSearch*, *Prefetch*, *Conventional*) as a fixed effect and Participant (nested within Group) and Task Set as random effects. Modeling Participant accounts for individual differences in performance and modeling Task Set accounts for any difference in the difficulty of the individual tasks. Note that mixed-model analyses can appropriately handle the imbalance in our data resulting from both groups using *Conventional*. We also performed post hoc pairwise comparisons using sequential Bonferroni corrections when applicable. Throughout this section, we also report least-squared means obtained from our mixed-model analyses.

5.1.1 Number of Tasks Completed

This metric measures overall performance. We expected task completion to increase with the *LocalSearch* and *Prefetch*. However, we found no significant difference in the *Number of Tasks Completed* using either of our asynchronous systems (2.67 using *LocalSearch* and 3.14 using *Prefetch*) and *Conventional* (2.84 tasks).

5.1.2 Round Trips

The conventional search process using standard web browsers proceeds as follows. First, a person issues a query to the search engine and then the server returns a search results page. This is one 'round' trip over the network. Then a person looks at the results and if he does not find the desired information he reformulates his query and requests new results incurring another round. If a page has embedded objects such as images or scripts, the browser automatically requests them from the servers resulting in another

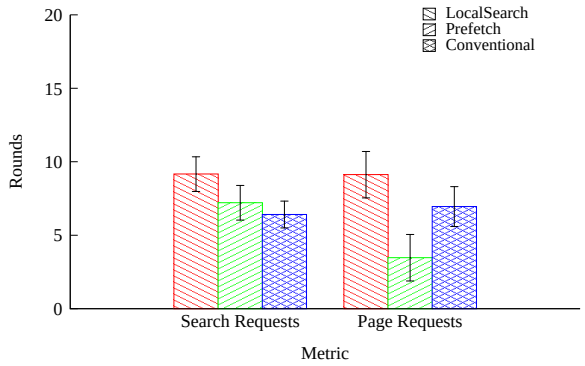

Figure 2: Number of *Round Trips* broken down into *Search Requests* and *User Initiated Page Requests*. Total time spent browsing is the same in all cases. Prefetch has fewer page request rounds.

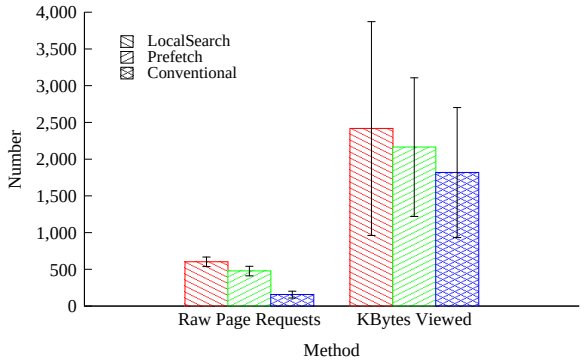

Figure 3: Number of *Raw Page Requests* and KBytes viewed per participant. More raw page requests and Kbytes viewed for LocalSearch and Prefetch than Conventional.

round of communication. While this process is not a problem for fast connections, when the connection is slow, each round incurs substantial idle time for the user. In the case of *Conventional*, the number of round trips is the sum of all *Search Requests* (this measures the search requests made by the user to the search engine) and *User Initiated Page Requests* (these are the requests made by the user explictly) that were not found in the cache. Images are disabled by default in *Conventional*, but scripts are often required for the complete rendering of a page. We give *Conventional* the benefit of not counting these against the number of page requests. Thus, the total number of requests is a lower bound on the true count. In the case of *LocalSearch* and *Prefetch*, the number of round trips consists of the sum of the number of remote *Search Requests* and remote *User Initiated Page Requests*.[2] We expected that the number of round trips incurred by these remote requests would decrease somewhat for *LocalSearch* and moreso for *Prefetch* as a result of users performing more local searches and browsing before issuing remote requests. Surprisingly, we found no significant effect of Method on the number of *Search Requests* (Figure 2: *LocalSearch* = 9.16, *Prefetch* = 7.21, *Conventional* = 6.41). We did find a significant effect of Method on the number of *User Initiated Page Requests* (*LocalSearch* = 9.12, *Prefetch* = 3.47, *Conventional* = 6.95, $F_{2,24.5} = 6.24, p \approx .006$). Post hoc pairwise comparisons show significantly lower *User Initiated Page Requests* experienced using *Prefetch* compared to *LocalSearch* ($F_{1,35.9} = 12.3, p \approx .001$) and *Conventional* ($F_{1,23.1} = 6.36, p \approx .019$). There was no significant difference between *LocalSearch* and *Conventional*.

5.1.3 Raw Page Requests

Raw page requests are the total number of pages requested by the user and the browser on the user's behalf (i.e. scripts, applets, etc.). This metric is meant to capture the amount of information requested and actually viewed by the user. For *Conventional*, the *Raw Page Requests* are simply all requests (both user initiated and browser initiated). For the asynchronous systems, the *Raw Page Requests* are similarly measured using only local requests (excluding UI frames). Remote requests are intentionally excluded from this metric because users may queue requests and not actually view them.

We found a significant effect of Method on the number of *Raw Page Requests* made by the browser on the user's behalf (*LocalSearch* = 601.9, *Prefetch* = 476.4, *Conventional* = 155.9, $F_{2,9.23} = 10.1, p \approx .005$). Figure 3 illustrates the *Raw Page Requests* along with KBytes of data actually viewed per participant for comparison. Post hoc pairwise comparisons show that the significant differences were between *LocalSearch* and *Conventional* ($F_{1,11.3} = 20.6, p \approx .0007$), and between *Prefetch* and *Conventional* ($F_{1,11.3} = 20.6, p \approx .0007$). This is presumably beneficial to the user, as more content is delivered with the asynchronous queueing interface per unit time.

5.1.4 Cache Hit Rate

The cache hit rate is the percentage of *Raw Page Requests* that were found in the proxy cache in *Conventional* or in the local proxy's cache in *LocalSearch* and *Prefetch*. In terms of the *Cache Hit Rate* of the number of objects, we found a significant effect of Method (*LocalSearch* = .099 (9.9%), *Prefetch* = .277 (27.7%), *Conventional* = .195 (19.5%), $F_{2,27.1} = 3.57, p \approx .042$) with the only significant difference between *LocalSearch* and *Prefetch* ($F_{1,30.4} = 7.23$, $p < .011$).

5.1.5 Bytes Downloaded

The number of bytes downloaded is measured as the total size of objects counted in the *Round Trips* metric. As expected, we found a significant effect of Method on *Bytes Downloaded* (Bytes) (*LocalSearch* = 864,735.9, *Prefetch* = 7,809,582.8, *Conventional* = 755,836.6, $F_{2,37} = 41.2, p < .0001$) with over 10 times more *Bytes Downloaded* by *Prefetch* compared to both *LocalSearch* and *Conventional*. The post hoc pairwise comparisons show that the signficant differences were between *Prefetch* and *Conventional* ($F_{1,37} = 71.6$, $p < .0001$), and between *Prefetch* and *LocalSearch* ($F_{1,37} = 50.8, p < .0001$).

5.2 Qualitative Results and Observations

Results from the exit questionnaire are summarized in Table 1. Of the *LocalSearch* group five people preferred the asynchronous system (50%), four preferred *Conventional*

[2]For these two metrics we only report results for the remote requests since they incur waiting time for the user.

Table 1: Exit Questionnaire Results Summary

Statement	% Agree
Prefer LocalSearch over Conventional	50%
Prefer Prefetch over Conventional	70%
Frustrated when using LocalSearch	30%
Frustrated when using Prefetch	20%
LocalSearch more effective than Conventional	50%
Prefetch more effective than Conventional	70%
Asynchronous system is not easy to use	20%
Text only browsing is useful	80%
Local search feature is useful	60%

(40%), and one person had no opinion (10%); for *Prefetch* the preferences were (70%), two (20%), and one (10%) respectively. Three people using *LocalSearch* reported being frustrated using the system (30%), and two people were frustrated when using *Prefetch* (20%). Half of our participants in group one felt that *LocalSearch* was more effective than *Conventional* (50%), and seven in group two felt that *Prefetch* was more effective than *Conventional* (70%). This is interesting because we found no statistically significant increase in task completion.

The three most common "best things" about either system were in reference to multitasking, queueing, and improved speed. By far the most common response to the "worst thing" about our system was the UI. Specifically, some comments were that it was "unfamiliar" and "confusing". That the screens used in this study were only 15 inches meant that the split frame format of our UI was severely detrimental to browsing space. We designed our UI assuming that the large screen from the formative study would be available. Other negatives were that the pages were sometimes broken and queued requests could be removed, but not interrupted after they started. Interestingly, only four out of the total twenty participants reported that the asynchronous system was not easy to use (20%). Sixteen participants out of twenty said that text only browsing was useful (80%), and twelve out of twenty found local search useful (60%).

6. DISCUSSION

We found that the *Number of Tasks Completed* exhibited no significant differences across conditions. This may be the result of including some subjective tasks in which completion was judged by participants themselves. In the future, including only objective tasks may be a better method of assessing task completion ability. Since the *Number of Tasks Completed* did not yield a significant result, we look at other metrics to understand the benefits and tradeoffs offered by local search and prefetching. We also discuss some of the realities and experimental considerations that led us to our study design and limitations.

6.1 Local Search

We found that there were more *Search Rounds* in *LocalSearch* than in *Conventional*. If however, we look at the components of *Search Rounds*, *Unique Search Requests* and *Unique User Initiated Page Requests*, there was no significant difference for either of these between *LocalSearch* and *Conventional*. This, along with no significant difference in the *Number of Tasks Completed*, shows that although local search does not improve the overall performance, it also does not worsen it.

One interesting finding was that the activity level (*Raw Page Requests*) of *LocalSearch* increased over three times compared to *Conventional*. Also, despite this, the difference in *Bytes Downloaded* was negligible. Since the total task completion was the same, this could indicate that people were able to request and view more information at no cost.

6.2 Local Search with Prefetching

Link prefetching is a well known method to use the idle time to download pages that are potentially useful to the user [26, 11]. The goal of this mechanism is to reduce the wait time for the successive page requests. Prefetching is beneficial for both low bandwidth and high latency environments because the number of round trips is reduced by downloading useful pages while the user is idle.

In our study the main difference we found was a reduction in the number of over the network requests for user initiated page requests with *Prefetch* compared to *LocalSearch* and *Conventional*. Our results indicated that 70% of users preferred *Prefetch* to *Conventional*. We argue that fewer round trips is a positive result particularly for network scenarios with higher latency than our own. Given a network configuration and a fixed epoch of time, only a limited number of network requests will be satisfied; as the latency increases (and bandwidth remains constant), fewer round trips per task implies that more tasks could be completed.

We also found that even with a slow connection up to 10 times more data could be downloaded with an asychronous prefetching system than a conventional browser. This is not perfectly true in our shared network environment, but this finding supports the use of prefetching for individuals with dedicated connections who do not compete for bandwidth.

6.3 Realities and Considerations

Due to the nature of our experiments being conducted in the field we encountered three practical difficulties and experimental design concerns which affected our study. Conducting the study in a controlled laboratory environment was not an option as there was no separate network available in the vicinity. While we would likely be able to control for some aspects of the study, the different environment would have taken away from the ecological validity of conducting the study in a realistic setting.

First, because the university network was in actual operation, we were not given direct access to the cache by the university's system administrators. Thus, we were not allowed to copy the cache and use a copy for each experimental condition and user (which would have been ideal). Simply using the cache directly would have caused contamination. Disabling caching altogether would bias comparison between the baseline and either of our systems. Given that the hit rate of our warmed cache was within the range of the observed cache hit rate at the university, we considered this to be an acceptable tradeoff.

Second, because school was in session, our participants could not uniformly commit to more than 90 minutes to the study (30 minutes per condition plus questionnaire and interview time). Given that the phenomenon of interest was slow connections, an inherent tension existed between setting the bandwidth throttle lower to observe more effects

and higher to accomodate user time constraints. We opted for the middle ground, but acknowledge that ideally, a longer running study with more tasks and lower bandwidth would yield better results.

Third, we would ideally compare idle time between the different systems, but measuring idle time directly was difficult for this study. We allowed multiple tabs in our study to allow users who preferred to multitask the freedom to do so. As a consequence, we found that stopwatch timing the screencapture of our participants was too imprecise, due to the significantly increased activity. Multiple tabs in conjunction with page download times and page rendering times make determining whether people are actually idle by automatically parsing activity logs error prone since it is unclear when pages are actually available. These are some of the practical reasons we decided to use the number of network *Round Trips* as a primary metric rather than idle time.

6.4 Limitations

It was not clear from our system level results whether the fact that the participants viewed more information is good in and of itself. It is possible that participants were viewing more pages, but not finding them relevant which could explain why task completion rates were the same despite this increase in information seen. However, during the experiment we did observe that participants using the asynchronous UI encountered several difficulties that could be improved. First, users were not familiar with the interface for our system. We believe that in the case of *LocalSearch* many users did not perceive enough of a benefit to prefer the asynchronous model over a more familiar interface (40%). However, with the inclusion of prefetching, 70% of users preferred the asynchronous system despite these UI flaws. Second, links on pages did not have an indication of whether they were in the cache. This resulted in people clicking on links and then having the additional step of manually queueing the page. Third, we found that the content pages in the cache were sometimes poorly rendered due to dynamic content including images, scripts, and style sheets that could not be downloaded by the prefetching process. We did not observe https page requests or http POST requests during our study.

Finally, prefetching is not free, it comes at the cost of additional bytes downloaded. Our results showed that the bytes downloaded by *Prefetch* increased by 10 times compared to *Conventional*. The bytes downloaded of *LocalSearch* only increased by an insignificant amount by comparison. In situations where the bandwidth is expensive, downloading additional pages indiscriminately results in additional cost to the user, but our simple breadth first search prefetching algorithm could be improved both in terms of overhead and hit rate by using proxy-initiated prefetching or other algorithms [11].

7. CONCLUSION

To our knowledge, this work is the first to present any detailed field study of web browsing behavior in the rural developing world context, and the consequences of applying an asynchronous queueing model along with basic web optimization techniques in these settings. Our study provides a detailed data point for a single constrained connectivity scenario. Our results should therefore impact future research in designing web search and browsing systems for these contexts. While there is room for improvement in our UI and algorithm (e.g. a UI with better screen space utilization for 15 inch screen), our participant feedback was largely positive. Our system level metrics indicate several performance improvments for the asynchronous systems. First, we found that queueing plus local search increased both the number of raw page requests and the bytes seen by the user per unit time. Second, we found that the inclusion of the prefetching mechanism reduced the number of round trips and increased the number of bytes downloaded per unit time. We hypothesize that in higher latency or intermittent network environments the synergy between queueing, local search, and prefetching is likely to have an even greater positive impact.

8. REFERENCES

[1] International World Wide Web Conference.

[2] S. Amershi and M. Morris. CoSearch: a system for co-located collaborative web search. In *Proceedings of CHI*, 2008.

[3] A. Balasubramanian, Y. Zhou, W. Croft, B. Levine, and A. Venkataramani. Web search from a bus. *Second Workshop on Challenged Networks(CHANTS)*, pages 59–66, 2007.

[4] BBC NEWS: FaceBook strips down to Lite site. http://news.bbc.co.uk/2/hi/technology/8249835.stm.

[5] E. Brewer, M. Demmer, M. Ho, R. Honicky, J. Pal, M. Plauche, and S. Surana. The challenges of technology research for developing regions. *Pervasive Computing, IEEE*, 5(2):15–23, 2006.

[6] P. Bruza, R. McArthur, and S. Dennis. Interactive Internet search: keyword, directory and query reformulation mechanisms compared. In *Proceedings of SIGIR*, pages 280–287, 2000.

[7] O. Buyukkokten, H. Garcia-Molina, A. Paepcke, and T. Winograd. Power browser: efficient web browsing for PDAs. In *Proceedings of CHI*, pages 430–437, 2000.

[8] J. Chen, L. Subramanian, and J. Li. RuralCafe: web search in the rural developing world. In *Proceedings of WWW*, pages 411–420, 2009.

[9] B. Du, M. Demmer, and E. Brewer. Analysis of WWW traffic in Cambodia and Ghana. pages 771–780, 2006.

[10] K. Fall. A delay tolerant network architecture for challenged internets. *Proceedings of SIGCOMM*, 2003.

[11] L. Fan, P. Cao, W. Lin, and Q. Jacobson. Web prefetching between low-bandwidth clients and proxies: potential and performance. In *Proceedings of the 1999 ACM SIGMETRICS international conference on Measurement and modeling of computer systems*, pages 178–187. ACM, 1999.

[12] S. Guo, M. Falaki, E. Oliver, S. Rahman, A. Seth, M. Zaharia, and S. Keshav. Very low-cost internet access using KioskNet. *ACM SIGCOMM Computer Communications Review*, 2007.

[13] S. Ihm, K. Park, and V. Pai. Towards Understanding Developing World Traffic. In *4th workshop on networked systems for developing regions (NSDR)*, 2010.

[14] S. Ihm, K. Park, and V. Pai. Wide-area Network Acceleration for the Developing World. In *Proceedings of the USENIX Annual Technical Conference*, 2010.

[15] S. Isaacman and M. Martonosi. The C-LINK System for Collaborative Web Usage: A Real-World Deployment in Rural Nicaragua. In *3th workshop on networked systems for developing regions (NSDR)*, 2009.

[16] M. Kamvar and S. Baluja. A large scale study of wireless search behavior: Google mobile search. In *Proceedings of CHI*, pages 701–709, 2006.

[17] M. Kamvar, M. Kellar, R. Patel, and Y. Xu. Computers and iphones and mobile phones, oh my!: a logs-based comparison of search users on different devices. In *Proceedings of WWW*, pages 801–810, 2009.

[18] A. Karlson, B. Bederson, and J. SanGiovanni. AppLens and launchTile: two designs for one-handed thumb use on small devices. In *Proceedings of CHI*, 2005.

[19] King, A. B. Website Optimization Secrets. 2008.

[20] A. Kumar, S. Agarwal, and P. Manwani. The spoken web application framework: user generated content and service creation through low-end mobiles. In *Proceedings of the 2010 International Cross Disciplinary Conference on Web Accessibility (W4A)*, pages 1–10. ACM, 2010.

[21] H. Lam and P. Baudisch. Summary thumbnails: readable overviews for small screen web browsers. In *Proceedings of CHI*, pages 681–690, 2005.

[22] Loband. http://www.loband.org.

[23] T. Loon and V. Bharghavan. Alleviating the latency and bandwidth problems in WWW browsing. In *Proceedings of USENIX Symposium on Internet Technology and Systems*, 1997.

[24] R. Luk, M. Ho, and P. Aoki. Asynchronous remote medical consultation for Ghana. In *Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*, pages 743–752. ACM, 2008.

[25] S. Mubaraq, J. Hwang, D. Filippini, R. Moazzami, L. Subramanian, and T. Du. Economic analysis of networking technologies for rural developing regions. *Workshop on Internet Economics*, 2005.

[26] V. Padmanabhan and J. Mogul. Using predictive prefetching to improve World Wide Web latency. *ACM SIGCOMM Computer Communication Review*, 26(3):22–36, 1996.

[27] G. Pass, A. Chowdhury, and C. Torgeson. A Picture of Search. In *First Intl. Conf. on Scalable Information Systems*, 2006.

[28] R. Patra, S. Nedevschi, S. Surana, A. Sheth, L. Subramanian, and E. Brewer. WiLDNet: Design and Implementation of High Performance WiFi Based Long Distance Networks. In *Proceedings of NSDI*, 2007.

[29] U. Pawar, J. Pal, and K. Toyama. Multiple mice for computers in education in developing countries. In *Proceedings of Intl. Conf. on Information and Communication Technologies and Development (ICTD), India*, 2006.

[30] A. Pentland, R. Fletcher, and A. Hasson. DakNet: rethinking connectivity in developing nations. *Computer*, 37(1):78–83, 2004.

[31] Port80 Software. http://www.port80software.com.

[32] A. Ratan, S. Satpathy, L. Zia, K. Toyama, S. Blagsvedt, U. Pawar, T. Subramaniam, and A. Ratan. Kelsa+: Digital Literacy for Low-Income Office Workers. In *Proceedings of Intl. Conf. on Information and Communication Technologies and Development (ICTD), India*, 2009.

[33] U. Saif, A. Chudhary, S. Butt, N. Butt, and G. Murtaza. Internet for the developing world: Offline internet access at modem-speed dialup connections. In *Proceedings of Intl. Conf. on Information and Communication Technologies and Development (ICTD), India*, 2007.

[34] H. Shen, M. Kumar, S. Das, and Z. Wang. Energy-efficient data caching and prefetching for mobile devices based on utility. *Mobile Networks and Applications*, 10(4):475–486, 2005.

[35] T. Sohn, K. Li, W. Griswold, and J. Hollan. A diary study of mobile information needs. In *Proceedings of CHI*, 2008.

[36] S. Surana, R. Patra, S. Nedevschi, M. Ramos, L. Subramanian, and E. Brewer. Beyond Pilots: Keeping Rural Wireless Networks Alive. In *Proceedings of NSDI*, 2008.

[37] W. Thies et al. Searching the world wide web in low-connectivity communities. 2002.

[38] H. Weinreich, H. Obendorf, E. Herder, and M. Mayer. Off the beaten tracks: exploring three aspects of web navigation. In *Proceedings of WWW*, pages 133–142, 2006.

[39] WiMAX forum. http://www.wimaxforum.org.

[40] S. Wyche, T. Smyth, M. Chetty, P. Aoki, and R. Grinter. Deliberate Interactions: Characterizing Technology Use in Nairobi, Kenya. 2010.

[41] X. Xie, G. Miao, R. Song, J. Wen, and W. Ma. Efficient browsing of web search results on mobile devices based on block importance model. In *Proceedings of Pervasive Computing and Communications*, pages 17–26, 2005.

[42] J. Yi, F. Maghoul, and J. Pedersen. Deciphering mobile search patterns: a study of yahoo! mobile search queries. In *Proceedings of WWW*, 2008.

[43] D. Zhang. Web content adaptation for mobile handheld devices. 2007.

[44] X. Zhang, J. Kurose, B. Levine, D. Towsley, and H. Zhang. Study of a bus-based disruption-tolerant network: mobility modeling and impact on routing. *Mobicom*, pages 195–206, 2007.