An Evaluation of Web Acceleration Techniques for the Developing World*

Amal Fahad† Zhuan Chen† Kai Shen† Jeffrey Bigham† Assmaa Fahad§
† *University of Rochester, U.S.A.* § *Baghdad University, Iraq*
{fahad, zchen, kshen, jbigham}@cs.rochester.edu, assmaa.fahad@gmail.com

Abstract

Web clients in the developing world experience long response times due to poor network connectivity. This paper evaluates existing and new web acceleration techniques using access traces from Iraq, Zambia, and Cambodia. We validate the effectiveness of proxy caching and observe around 40% cache hit ratio. But our evaluation finds limited benefit of web prefetching (<1% hit rate by the history-based prefetching that previously reported around 10% hit rate [7]). This is because proxy caching diminishes the benefits of prefetching at individual clients and further many (40–60%) cache misses are accesses to dynamic web applications that are not prefetchable. To accelerate dynamic web applications for the developing world, we explore the feasibility and performance of mirroring dynamic applications on near-client proxy servers. Preliminary evaluation on a collaborative office application shows an order of magnitude performance improvement by the local mirroring.

1 Introduction

Developing countries like those in Sub-Saharan Africa and war-torn Middle East have poor network connectivity. Their global Internet access is bandwidth-constrained, sometimes based on long-latency satellite links. Due to poor local infrastructure, high-speed cable networks are rare and users are often connected by lower-speed, less reliable, point-to-point wireless links [14]. Figure 1 illustrates the network infrastructure for the Baghdad University and the neighboring district at the southern part of the city of Baghdad (Iraq).

Web clients in these regions must endure excessively long request response time. As an example, the start-up access to Google Docs (with cold browser cache) takes over 20 seconds through a computer at the College of Science in Baghdad University. Faculty and students there have to explicitly pass updated documents in collaborative editing despite the availability of relevant web-based services. Such information technology challenges, called by many the "global digital divide" [17], have hampered the economic, societal, and educational development that these regions urgently desire.

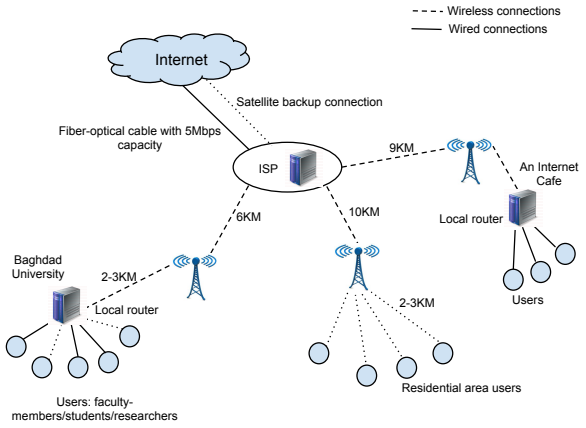

Figure 1: Network infrastructure for the Baghdad University and the neighboring Al-Saydiya district at southern Baghdad.

Past studies [3, 6, 12] analyzed the web access traffic and usage pattern at the developing world. Previous work [3,7,11,13] also suggested that web prefetching can improve the performance for clients with slow network accesses. Offline browsing [3, 6] enables the browsing of cached content without the network connectivity. The network data volume can be further reduced through content de-duplication [10].

However, limited attention has been paid on the issue of dynamic web applications for the developing world. Unlike static HTTP file transfers, dynamic applications that execute code to produce response content are rapidly dominating the web. Dynamic web requests are often unsuitable for caching, prefetching, or offline browsing due to their dependence on fast changing data on the server. This paper characterizes dynamic web requests in developing world web accesses and evaluates their impact on existing web acceleration techniques. Motivated by the success of proxy caching for static requests, we propose a new technique to host a mirror copy at a near-client proxy server for dynamic applications.

2 Web Trace Characterization

We analyze web performance characteristics in the developing world using real user traces. We captured a web

*This work was supported in part by the National Science Foundation (NSF) grants CCF-0937571 and CCF-0448413. Shen was also supported by a Google Research Award and an IBM Faculty Award.

trace in Iraq with the help of the Internet Service Provider that serves Baghdad University and nearby residential districts. Further, authors of two prior studies [6,12] have kindly shared traces from Zambia, captured at the gateway that connects the satellite to the wireless network that covers a rural area, and Cambodia, collected at the Cambodia Communication Information Center, with us. The Iraq, Zambia, and Cambodia traces were collected in 2011, 2010, and 2005 respectively. They contain two, four, and eleven million web accesses respectively.

Proxy Caching All three traces were collected on web proxy servers with caching support, which allows us to examine the performance of proxy caching. Figure 2(A) shows the observed web cache hit ratios. For cache hits, we distinguish those that require a content freshness check (HTTP `If-Modified-Since`), which incurs a network round-trip to the original server on the Internet. Results show high cache hit ratios (40% or more for all three web traces). Figure 2(B) shows the average web response time in each case. Cache hits lead to about two orders of magnitude reduction in the response time. Even cache hits with freshness check result in about one order of magnitude speedup. These results validate the high effectiveness of proxy caching.

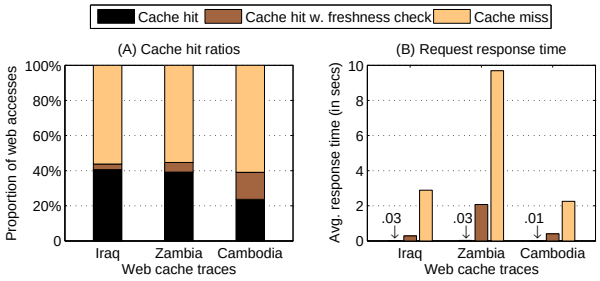

Figure 2: Web proxy caching performance.

Dynamic Web Applications Dynamic applications that execute code to produce response content are increasingly common on the web. These requests are often uncachable because it is difficult to check the freshness of previously generated dynamic response. We identified dynamic web requests in our traces by recognizing certain patterns in the requested URL. First, only dynamic requests carry parameters in the form of a query component in the URL [1]. We recognize the existence of a query component when capturing special characters of '?' or '=' after the URL path component. Second, parameterless dynamic web requests do not possess the query component in their URLs. We attempt to identify some by recognizing dynamic request file extensions (currently '.php' and '.asp').

We find that, among cache misses, the proportions of dynamic web accesses are 59%, 62%, and 41% for the Iraq, Zambia, and Cambodia traces, respectively. This result shows that dynamic web accesses are a primary contributor to cache miss requests, and even more so in the recent Iraq/Zambia traces (2011/2010) than in the Cambodia trace (2005). The significance of dynamic web accesses and the inability of existing techniques (including caching) to address them present a critical challenge for the developing world web performance.

At the same time, we find that the dynamic web requests concentrate on a few popular sites. We recognize the target site of a dynamic web access as the authority component of the URL (the segment preceded by '//' and terminated by '/' or '?' [1]). With thousands of accessed dynamic application sites, Figure 3 shows that a few most popular sites account for a significant proportion of total dynamic web accesses. In particular, the top eight sites receive 30%, 44%, and 37% of all accesses for the three traces respectively. As a result, a large benefit could be achieved by accelerating only a few important web applications to the developing world clients.

3 Web Prefetching Effectiveness

Predicting future web accesses and prefetching them ahead of user requests may reduce user response time. While high-speed networks in the developed world have practically eliminated the utility of prefetching, its potential benefits are still critical for developing world clients with long-latency, low-bandwidth Internet accesses. In this section, we evaluate the effectiveness of prefetching techniques for developing world using the following metrics: (1) *hit rate*—the number of hits to the prefetching cache divided by the total number of accesses; (2) *overhead rate*—the number of downloaded pages due to prefetching divided by the number of accesses; (3) *use rate*—the proportion of prefetched pages that are indeed accessed later.

Recent studies [3, 11] have employed prefetching to accelerate web accesses for developing world clients. Specifically, Isaacman and Martonosi [11] designed a folder-level prefetching strategy to prefetch all directory contents upon requesting some of its components. However, the data-mule model they used for data downloading assumes that the bandwidth is unlimited at the time of prefetching. Such a model does not apply to many developing world sites with permanent (but slow) Internet connections. Chen *et al.* [3] used Mozilla Fasterfox to prefetch pages in anchor links that appear in recent downloaded pages. They achieved less than 2% hit rate with 22% overhead rate.

Although they are easy to implement, these developing world prefetching techniques come with a modest benefit that can be attributed to the simplicity of their prefetching heuristics, and particularly by not taking advantage of user access pattern history. Such history is

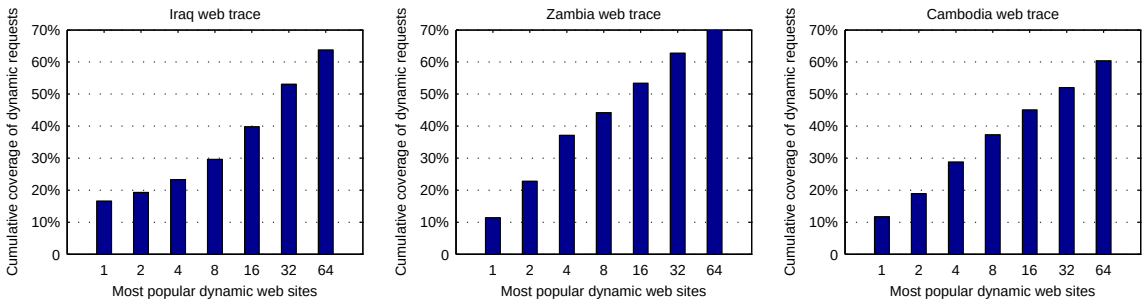

Figure 3: Proportion of dynamic requests that belong to a few most popular application sites. There are a total of 6,772 accessed dynamic sites in the Iraq trace; 7,959 sites in the Zambia trace; and 13,668 sites in the Cambodia trace. Facebook, Gmail, and Yahoo! Search were the number one application in each of the three traces.

readily available at a local proxy cache server where our three web access traces were collected. History-based prefetching techniques [7, 13] learn user's web surfing behavior in order to predict future accessed pages. In particular, Fan *et al.* [7] used the Prediction-by-Partial-Matching (*PPM*) method to build history trees. Each tree predicts a sequence of target URL accesses to follow another sequence of predictor URL accesses. Their evaluation showed that PPM can achieve up to 10% hit rate with 23% overhead rate when predicting 4-URL targets using 2-URL predictors. Such performance appears very desirable and therefore we evaluate PPM using our developing world web traces.

Evaluation of the PPM Prefetcher Our evaluation uses the Zambia and Cambodia web traces. The Iraq web trace does not distinguish users (all requests carry the same ID) which prevents us from building per-user access sequence. This table shows the performance results of our trace-driven evaluation of the PPM prefetcher.

Traffic location	Hit rate	Overhead rate	Use rate
Zambia	0.2%	2.5%	1.7%
Cambodia	0.2%	8.7%	1.1%

Our results are disappointing—0.2% hit rate is much less than what was promised in earlier work [7]. A key reason is that much of the benefit of the history-based prefetching has already been realized by the proxy web cache. Specifically, the prefetching benefit relies on captured patterns of past accesses (mostly from other users). In a proxy server that caches past accesses from a large number of users, those "predicted" accesses are probably already in the cache. Earlier history-based prefetching work [7, 13] targets dial-up modem users for which no near-user proxy server could serve a large number of users—if it is near user, it is at the user side of a modem link and thus it serves one user. Today proxy caching servers are commonplace, even for the developing world (all three of our developing world web traces were collected at proxy servers with caching support).

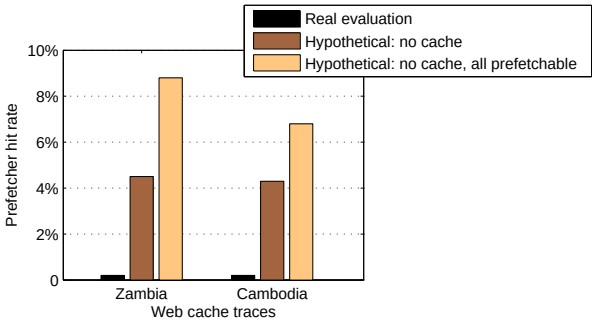

Figure 4: The PPM prefetcher hit rate. Results include the real evaluation result and two hypothetical situations—the first assumes no proxy cache, the second assumes no proxy cache and that all pages are prefetchable.

Another reason for the low prefetch hit rate is that the non-prefetchable dynamic web requests are increasingly dominant in today's web accesses (evaluated in Section 2). They reduce the room of applicability for classic web prefetching techniques.

To illustrate how these factors have diminished prefetching effectiveness, Figure 4 depicts the PPM prefetcher hit rates for the real evaluation and two hypothetical situations. The first hypothetical situation assumes no proxy cache and its difference with the real evaluation shows that the effectiveness of proxy web caching substantially diminishes the benefits of prefetching. This is bad news for prefetching, but not for users who do not care whether the fast response is due to caching or prefetching. Our second hypothetical situation in Figure 4 assumes no proxy cache and that all pages are prefetchable. For the latter assumption, we simply divide the hit count by the number of prefetchable accesses (as opposed to dividing by all accesses in the real evaluation). The different hit rates between the two hypothetical situations show that the reduced amount of prefetchable accesses (largely due to the dominance of dynamic web requests) further contributed to the decline of prefetching effectiveness.

4 Local Application Mirroring

Our analysis so far reaches two key conclusions—1) proxy caching of static web pages is very effective for improving the developing world web performance; 2) additional improvement requires new technique to address dynamic web applications. In such a context, we speculate for a high performance gain if caching can be applied to dynamic web applications. Such caching will substantially improve dynamic web request response time and reduce data traffic on critical Internet access links for the developing world. This caching concept is also supported by our evaluation in Section 2 that shows a few important dynamic web applications account for a large proportion of total dynamic web requests.

Caching dynamic web applications requires a critical system support, we call *local mirroring*, of hosting a local copy of dynamic web applications on near-client proxy servers. While local application mirroring will introduce new challenges in data consistency, deployment and security, many existing techniques (particularly in the area of cloud computing) can be leveraged to accomplish it. We discuss several related issues in this section, and also provide a preliminary evaluation of a web-based collaborative office application to show the feasibility and potential benefits.

In related work, near-client web services are also supported by edge computing platforms like Akamai and Amazon CloudFront that are hosted at established Internet Points-Of-Presence. The reach of edge services is still very thin in the developing world. Researchers have also started customizing web applications for the developing world clients. For instance, the RuralCafe project [4] modified a search engine application such that high-client-interactivity functions are kept locally while the remaining work is performed at the remote server. These efforts require significant customization and programmer work to web applications. Instead, we explore system-level support of local application mirroring with minimum application changes.

4.1 Application Data Management

Since application data is hosted on both a local mirror and the global application site, it raises the concern of distributed data management. Some dynamic applications run with read-only state or soft (cached) state. For example, content transformation applications (like converting web content to speech for people with reading difficulties [2]) typically contain only cached state (such as the text-to-speech transformed audio files). Local application mirroring allows these applications to be replicated at multiple local sites to achieve better experience of interactivity in each local region, where each site uses its own state separately from others without introducing the problem of state accessibility and consistency.

Dynamic applications that are stateful usually store data on a global site (probably in a cloud computing center) in order to obtain the benefits of accessibility, availability, consistency, and data sharing for better user interaction experience. Examples include applications that support collaborative work within a group [8] and social network applications in which users post and share messages, images, and videos. These applications require a high level of network interaction among clients. While local application mirroring is expected to realize such requirement within the local region, the global state accessibility and consistency presents new challenges when some of these applications' data are also stored locally.

A successful solution to this problem must rely on the access locality. For collaborative work within a local group, it is common that data generated locally are likely to be consumed locally as well. A sample from Baghdad University shows that people's friends in Facebook exhibit strong locality (about 73% of their friends are located within the same local region). Such friendship locality suggests strong local interaction and collaboration for many other web-based applications as well. Furthermore, because the network access is limited in the developing regions, a user tends to use network access from a single place (e.g., university office), thereby reducing the chance for a single user to access web applications from multiple local networks. The evidences of service access locality lead us to emphasize the optimization of interaction within the local region.

We propose to have multiple copies of an application to be hosted on different locations based on users' demand (in the style of on-demand caching). Maintaining data consistency across multiple copies can be done in a hierarchical fashion. The interaction between local users are handled by the near-client proxy server to guarantee high-quality service experience, which further combine and propagate their updates to the global site periodically. The global site treats the updates from a near-client proxy server as regular client updates. Conflicts may still exist if simultaneous accesses to the same data partition are made from multiple local sites. The global site will resolve these conflicts like it handles simultaneous conflicts from multiple regular clients. Note that the chance of such conflicts should be rare if our expectation of strong access locality (mentioned above) holds.

4.2 Deployment and Security

Local deployment for applications that originally run on a central cloud imposes new challenges such as compatibility and security. Today's web applications are developed by various programming languages and techniques. Locally hosting the applications from different

service providers requires the local platform to meet all their dependences including the runtime environment, libraries, configurations, and so on. The diversity of global and local platforms further complicates the deployment and maintenance, which demand expert knowledge that is often unavailable in developing regions. Furthermore, hosting multiple applications on one local server introduces potential security concerns including the maintenance of application privacy and the containment of security vulnerability.

Native machine-level virtualization (*e.g.*, VMware and Xen), which is widely used in cloud computing, can be leveraged to the local application mirroring. It presents the virtual machine (VM) interface to encapsulate the full application/system ensemble and provide effective migration mechanisms between hosts. Virtualization enables strong isolation among applications and also allows applications with diverse security requirements to be run concurrently on one platform [9]. Based on the easy deployability provided by virtualization, we suggest to expose the near-client proxy server as a local cloud platform with VM-based interfaces. Virtualized cloud services can be distributed as is. Alternatively, a service provider may generate an adaptive version of their applications for local mirroring. Such adaptive application mirror can customize to local user preferences, restricted security environments, or constrained bandwidth for installation and operation.

Unlike the VM migration in clouds where the fast and stable network connection exists, VM-based deployment reaching the developing world must optimize for low-bandwidth network connections. While native VM images are typically large (725 MB for our Xen image with a Feng Office installation [8]), VM images for web applications share many system and library components and therefore each contains a relatively small amount of unique content. With a stock image at a local proxy server, the deployment of a new VM image only requires the delta between the new and stock images (43 MB for the Feng Office installation identified through `rsync` [15]). More sophisticated content redundancy elimination techniques [16] will also be helpful.

Additional security concerns may exist for local application mirroring. Compared with the third-party clouds (like Amazon EC2), which provide the security guarantee under contracts, near-client proxies are located and controlled totally by local sites without strong restraint. The less guarantee of privacy, integrity, and availability, due to the weaker resource and technical support, makes the local platform potentially less trusted. Based on the virtualized infrastructure, extra security enhancement can be added underneath the virtual machine monitor (VMM) to provide additional privacy protection and integrity checking to the hosted VM [18].

4.3 Evaluation of A Collaborative Application

We provide a preliminary evaluation to demonstrate the benefit of local application mirroring. We use Feng Office [8] as an example of group collaboration application that supports web-based office document editing and sharing. Our tests used a workload that simulates collaborative editing using published revision history of Wikipedia documents, specifically, the "Harry Potter" article. Typically, read operations are much more frequent than writes in a mixed read/write scenario. Thus we use a 9:1 read/write ratio in our workload.

We compare the performance between accessing a globally hosted application and accessing a locally mirrored application for developing world users. Given the relatively low-end machines used in the developing world, we also want to see whether the restricted resource would become a potential bottleneck under high client loads. We evaluated the performance of Feng Office in two scenarios: the near-client proxy server (represented by a low-end machine with a single-core 2.66 GHz Pentium 4 processor and 512 MB memory) and the remote web server (represented by a high-end machine with two dual-core (four cores total) 3.00 GHz Xeon CPUs and 4 GB DRAM memory).

To emulate the performance measurement of the remote Internet sites and the network access conditions from the developing world, we ran the Linux traffic-control tool (Tc) on the high-end machine and configured its network bandwidth and latency to the estimated value for the target developing world locations. Specifically, our network performance estimation uses the response time of cache miss requests in the Iraq, Zambia, and Cambodia traces mentioned in Section 2. Our estimated network latency is 1.48, 6.03, and 1.77 seconds for the three sites respectively. Our estimated bandwidth is 106, 196, and 208 Kbps respectively.

Figure 5 shows the evaluation result. For low request concurrency, the average response time of Feng Office running on the near-client proxy server is one order of magnitude shorter than that on remote Internet. On remote global sites, the long transmission time (due to the large network latency) dominates the request service period, especially for the case of Zambia.

The weaker CPU of the local server could be a bottleneck as the load rises (higher number of concurrent requests). While the benefits of near-client proxy are affected by this, results show that the local server still provides better response time compared to the remote global server for up to 64 concurrent requests. We also note that the cloud offloading techniques employed for resource-constrained computing (as in MAUI [5]) can be leveraged to help alleviate the resource constrains at local proxy servers.

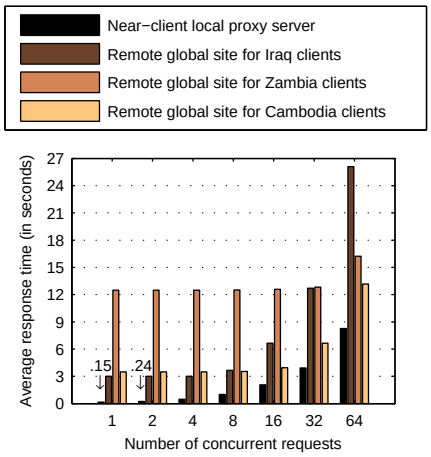

Figure 5: Performance comparison of the near-client proxy server and remote server for Feng Office.

We also observe the somewhat surprising result that the global site performance for Iraq clients degrades more sharply than the local proxy server performance does when the concurrent request rate goes up. This is due to high contention on the developing world Internet access bandwidth (lowest for the Iraq site among our three sites). This shows that our local application mirroring can provide substantial benefit even at high load.

5 Conclusions

This paper evaluates developing world web acceleration techniques using traces from Iraq, Zambia, and Cambodia. We validate the effectiveness of proxy caching but we find limited benefit of web prefetching ($<$1% hit rate by the history-based prefetching that previously reported around 10% hit rate [7]). This is because proxy caching diminishes the benefits of prefetching at individual clients and further many (40–60%) cache misses are accesses to dynamic web applications that are not prefetchable. To accelerate dynamic web applications, we propose the local application mirroring to host network applications on local sites nearby their users. Preliminary evaluation on a collaborative office application shows an order of magnitude performance improvement by the local mirroring. We also discuss its challenges in terms of data consistency, deployment and security. We are hopeful that a combination of existing techniques (particularly in cloud computing) and new solutions will eventually overcome these challenges.

References

[1] T. Berners-Lee, R. Fielding, and L. Masinter. Uniform resource identifiers (URI): Generic syntax. Network Working Group, Internet Engineering Task Force, Aug. 1998. http://www.ietf.org/rfc/rfc2396.txt.

[2] J. P. Bigham, C. M. Prince, and R. E. Ladner. Addressing performance and security in a screen reading web application that enables accessibility anywhere. In *8th Int'l Conf. on Web Engineering (ICWE)*, 2008.

[3] J. Chen, D. Hutchful, W. Thies, and L. Subramanian. Analyzing and accelerating web access in a school in peri-urban India. In *20th Int'l Conf. on World Wide Web (WWW)*, 2011.

[4] J. Chen, L. Subramanian, and J. Li. RuralCafe: Web search in the rural developing world. In *18th Int'l Conf. on World Wide Web (WWW)*, 2009.

[5] E. Cuervo, A. Balasubramanian, D.-K. Cho, A. Wolman, S. Saroiu, R. Chandra, and P. Bahl. MAUI: Making smartphones last longer with code offload. In *8th Int'l Conf. on Mobile Systems, Applications, and Services (MobiSys)*, 2010.

[6] B. Du, M. Demmer, and E. Brewer. Analysis of WWW traffic in Cambodia and Ghana. In *15th Int'l Conf. on World Wide Web (WWW)*, 2006.

[7] L. Fan, P. Cao, W. Lin, and Q. Jacobson. Web prefetching between low-bandwidth clients and proxies: potential and performance. In *ACM SIGMETRICS*, 1999.

[8] Fengoffice. http://www.fengoffice.com.

[9] T. Garfinkel, B. Pfaff, J. Chow, M. Rosenblum, and D. Boneh. Terra: A virtual machine-based platform for trusted computing. In *19th ACM Symp. on Operating Systems Principles (SOSP)*, 2003.

[10] S. Ihm, K. Park, and V. S. Pai. Wide-area network acceleration for the developing world. In *USENIX Annual Technical Conf.*, 2010.

[11] S. Isaacman and M. Martonosi. Potential for collaborative caching and prefetching in largely-disconnected villages. In *ACM Workshop on Wireless Networks and Systems for Developing Regions (WiNS-DR)*, 2008.

[12] D. L. Johnson, V. Pejovic, E. M. Belding, and G. van Stam. Traffic characterization and internet usage in rural Africa. In *20th Int'l Conf. on World Wide Web (WWW)*, 2011.

[13] V. N. Padmanabhan and J. C. Mogul. Using predictive prefetching to improve World Wide Web latency. *ACM SIGCOMM Computer Communication Review*, 1996.

[14] 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 *4th USENIX Symp. on Networked Systems Design and Implementation (NSDI)*, 2007.

[15] A. Tridgell. *Efficient Algorithms for Sorting and Synchronization*. PhD thesis, The Australian National University, 1999.

[16] T. Wood, K. Ramakrishnan, P. Shenoy, and J. Van der Merwe. CloudNet: dynamic pooling of cloud resources by live WAN migration of virtual machines. In *7th ACM Conf. on Virtual Execution Environments (VEE)*, 2011.

[17] World summit on the information society (Geneva 2003—Tunis 2005). What's the state of ICT access around the world? http://www.itu.int/wsis/tunis/newsroom/stats/.

[18] F. Zhang, J. Chen, H. Chen, and B. Zang. CloudVisor: Retrotting protection of virtual machines in multi-tenant cloud with nested virtualization. In *23th ACM Symp. on Operating Systems Principles (SOSP)*, 2011.