# Opportunistic Web Access via WLAN Hotspots

Mikko Pitkänen[1], Teemu Kärkkäinen[2], and Jörg Ott[2]
[1]Helsinki Institute of Physics, Technology Programme <mikko.pitkanen@hip.fi>
[2]Aalto University School of Science and Technology, Department of Communications and Networking

***Abstract*—Mobile phones are becoming commonplace for consuming Internet content and services. However, availability, affordability, and quality of the supposedly ubiquitous cellular network infrastructure may be limited, so that delay-tolerant web access via WLAN hotspots becomes an interesting alternative, even in urban areas. In this paper we explore mobile web access using asynchronous messaging via WLAN hotspots: for nodes directly connected to an access point and nodes relying on others for message forwarding. We investigate different routing and caching approaches using real-world access point locations in Helsinki. We find that a significant number of requests can be satisfied without requiring an always-on infrastructure, provided that users are willing to tolerate some response delay; this allows offloading traffic from the cellular network. We also report on our prototype implementation of mobile DTN-based web browsing.**

## I. Introduction

Mobile devices and particularly mobile phones gain increasing importance as primary means to access content and services on the Internet. This is achieved by taking advantage of the seemingly ubiquitous connectivity offered in urban areas by today's cellular operators. However, in practice pervasive mobile access over cellular is hampered in numerous ways.

First, cellular coverage may not be as ubiquitous as expected, as even in well-provisioned countries and cities it is not unusual for calls or data connection to get disrupted. Second, cellular access may be undesirable due to the high costs associated with roaming charges or traffic volume based pricing models. Finally, even if cellular connectivity is *theoretically* available and affordable, there are *practical* limitations in the current networks. For example, heavy queuing can be used inside the network, e.g., to reduce data loss on the wireless link. This leads to significant round-trip times that impact the user experience especially when accessing large web pages comprising many objects [5].

Moreover, the current trend of gearing mobile handsets towards Internet usage has lead to devices such as the *iPhone* with data-hungry applications assuming and using always-on connectivity at rapidly increasing rates. The result is increasing congestion and noticeably reduced performance of cellular networks[1], especially when users are forced to compete for the shared communication resource without guaranteed bandwidth allocations as was observed in cellular networks [5].

The many commercial WLAN hotspots spread across urban areas and an increasing number of WLAN community networks present an interesting opportunity for Internet access. While they may not be as thoroughly administered as cellular networks, they offer sizable access rates, exhibit short RTTs, and are usually underutilized. This makes them a suitable platform for mobile web access, though with the caveats that 1) their coverage is very limited and 2) access is usually constrained to subscribers or community members.

In this paper we overcome both of these limitations by using opportunistic delay-tolerant networking to extend the reach of WLAN hotspots to users that are not authorized to directly access them or are out of their radio range.

While using mobile ad-hoc networks to indirectly access the Internet is not new (see, e.g., [3], [17]), in practice the potentially sparse node population and the many round-trips to retrieve all objects from web pages[2] may lead to high delays or prevent successful retrieval altogether, e.g., due to path instability. Therefore, we borrow the concept of asynchronous messaging from Delay-Tolerant Networking (DTN) [10] that enables communication even without an instant end-to-end path. We expand our earlier work on web access via DTN [20], which reduces the typically verbose HTTP interactions to a single round-trip and makes web access suitable for DTNs, to cover a combination of opportunistic networking and hotspots. For further optimizations, we exploit the caching friendliness of the message-oriented communications in DTNs. We investigate the impact of opportunistic caching in both access points and mobile nodes by expanding our earlier work on cooperative caching in mobile ad-hoc networks [24].

This approach largely solves issue 1) for users who are able to relax their delay constraints on receiving a response by waiting until an access point comes into reach or until messages can be opportunistically relayed through other nodes to one. The latter also helps addressing issue 2) as network access becomes indirect and is delegated to other users; this is feasible because of common "flat rate" tariffs and the high access rates which are usually not exhausted

[1]"Customers Angered as iPhones Overload AT&T", The New York Times. http://www.nytimes.com/2009/09/03/technology/companies/03att.html

[2]For 2007, an average web page was reported to comprise some 50 objects and a total data volume of more than 300 KB; see http://www.websiteoptimization.com/speed/tweak/average-web-page/, 2009.

by the subscriber's device alone.

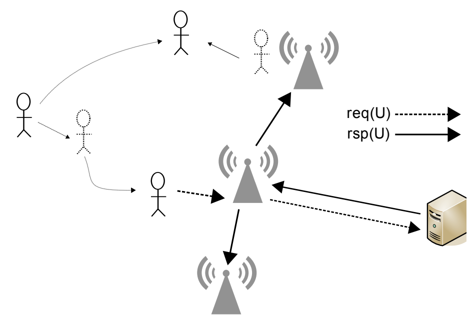

Figure 1. Opportunistic content access via WLAN hotspots.

Figure 1 illustrates a sample communication scenario consisting of several mobile users in an urban area. The mobile users' devices are capable of communicating among each other opportunistically when they come into radio range. When in the coverage of a WLAN hotspot, the mobile nodes connect to the Internet via the access point that provides DTN routing and, optionally, additional caching functionality. Geographically neighboring WLAN access points may further conspire to replicate responses from the Internet to reach (indirectly connected) mobile nodes who may have moved in the meantime. Mobile nodes and access points holding a copy of the sought content may reply immediately to requests. This yields a DTN messaging overlay among the mobile nodes, the hotspots, and the Internet, that enables mobile users to choose when to use delay-tolerant and when cellular Internet access.

In effect, this offloads data belonging to delay-tolerant interactions from the 3G network to the closest WLAN access points, thus helping to preserve cellular connectivity for real-time and more interactive applications. Several applications, for example, fetching RSS feeds, P2P file retrieval, or posting to blogs do not require immediate feedback, and thus are good candidates to benefit from the presented opportunistic web access model. Obviously, the cellular network can also serve as a backup if DTN-based communication does not yield a result in the expected time frame. Altogether, complementary DTN overlays support and enhance pervasive information access for mobile users.

Our main contributions are threefold. 1) We design a system architecture for DTN-based web access using WLAN hotspots. 2) We confirm the feasibility of our design by means of an extensive simulation study using a map-based mobility model with map data, hotspot locations, and web pages taken from the real world. 3) We validate the practical feasibility of our concepts in an implementation including a native DTN web server, an access point supporting DTN routing and caching, and mobile web applications for Android, the iPhone, and (via a local proxy) for Linux and Mac OS.

After reviewing related work in Section II, we present our DTN-based content access model in section III. In Section IV, we report on our simulation findings to show the applicability of the model and in Section V we present our implementations. We conclude this paper in Section VI with a brief assessment and discussion and hint at future work.

## II. Related Work

Mobile Internet access—directly via diverse wireless technologies or indirectly via cooperating peers —has been a research topic for more than a decade. Various approaches for dealing with opportunistic connectivity between mobile nodes and to access networks can be found.

Numerous proxy architectures were developed for Internet access in disconnected environments. Drive-thru Internet [20] suggests a proxy-based solution for Internet access in disconnected environments by gatewaying between interactive HTTP request-response sequences and asynchronous DTN communications. In this paper, we re-use our earlier scheme for encapsulating HTTP in DTN messages. The Tetherless Computing Architecture [28] has pursued similar ideas.

Besides content retrieval, searching for content in disconnected mobile environments has received quite some attention. Balasubramanian et al. [2] presented *Thedu* to enable efficient web search from a city bus for mobile nodes with intermittent WLAN connectivity: search queries collected by clients are aggregated by a central search engine, and the most relevant results are prefetched. Thedu assumes intermittent, but direct connectivity to a WLAN through which the server can be accessed and places its emphasis on dealing with search results rather than on generic page requests as we do. Earlier work by the authors has investigated search request forwarding strategies for finding content in mobile opportunistic networks with and without Internet gateways [23]. The Haggle system [26], [19] provides an architecture for sharing and retrieving content from within (mobile opportunistic) social networks, associating users with social context for routing/forwarding and content with metadata for searching and identification.

In addition, various systems for pro-active content dissemination were developed. TACO-DTN [29] assumes an infrastructure backbone made up of well-connected infostations, e.g., part of travel information system attached to bus stops. The infostations are used as gateways to publish contents to which the mobile nodes subscribe when they come into range. Published content is routed to those infostations with active subscriptions to be stored and ultimately delivered to the mobile nodes. A system for urban content dissemination originated from fixed access points is analyzed in [16]. A publish/subscribe architecture or large-scale (rural) dissemination is presented in [12]. PodNet [14] describes an interest-based content sharing system between mobile nodes in which content is organized according to channels described by metadata.

Finally, efficient content access can be supported by cooperative caching inside the network and in mobile nodes, reducing the distance between requester and a copy of the content and enabling content access without the need for infrastructure access. Internet access via densely populated MANETs was studied (e.g., [17]) and numerous approaches to caching were devised: Lin and Cao applied caching in MANETs using redirection of request packets in intermediate systems towards areas with cached copies [33]. Ditto [8] is a caching system for multi-hop wireless networks that divides content into *chunks* of data that can cached in the nodes along the data path and in those overhearing the wireless transmissions. Afanasyev et al. [1] propose caching for individual packets at the link layer of wireless networks; nodes overhearing wireless transmissions can keep packets to satisfy later requests.

Our work differs from the above in at least one of three key respects: we do not operate at the level of individual packets but larger messages, we aim at supporting the traditional access methods to structured web contents, and we focus on opportunistic networks. Our earlier work [24] has investigated the effect of opportunistic caching on content retrieval performance in mobile ad-hoc networks in which some nodes offer direct Internet connectivity. We borrow the idea of caching for optimization and expand it to include caching in both fixed and mobile nodes; we also move beyond our originally simplistic assumptions and model message content and web retrieval more accurately based on real-world hotspots and web sites.

## III. Opportunistic Web Access

Web content is accessed by sending one HTTP request per web resource and receiving a corresponding response with a message body containing the requested object. As typical HTML web pages comprise multiple objects such as images, a series of request-response pairs is required per page to retrieve all the embedded objects, leading to a retrieval delay depending on their number, size, and the round-trip time, among other factors.[3]

This kind of iterative retrieval is only practical if the endpoint is continuously connected to the Internet, otherwise retrieval of some objects will fail and the requested page cannot be fully displayed. Furthermore, low round-trip times are required for acceptable retrieval performance or the iterative process will take too long, e.g., in the order of minutes. [5] Finally, while HTTP is stateless and could, in principle, support asynchronous communications, it relies on TCP and thus requires end-to-end connectivity to the origin server (or a cache/proxy) for the duration of a transaction, i.e., the retrieval of (parts of) a single object, preventing asynchronous access via intermediate nodes. [20] A direct end-to-end path also ensures that responses are properly routed back to the requesting node (that may be tracked using IP or transport layer mobility mechanisms if it moves while retrieving).

To overcome the above limitations and enable DTN-based web access as depicted in figure 1, directly from within a hotspot as well as indirectly via other mobile nodes, we have to reduce interactivity and eliminate the need for an end-to-end path (section III-A) while maintaining (indirect) mutual reachability of the mobile node and the hotspots for delivering requests and responses (III-B). We finally discuss optimizations by means of resource caching (III-C) to improve retrieval performance even under unfavorable conditions.

### A. DTN-based Resource Retrieval

We follow the Delay-tolerant Networking architecture [4] and use its Bundle Protocol [27] as the basis for asynchronous message-based communications which eliminates the need for end-to-end paths. Nodes are typically represented as *Bundle Protocol Agents* with which application instances register to send and receive messages. Communication between bundle agents uses *convergence layers* that provide a mapping to specific links or networks, e.g., to TCP [7] for building overlays across the Internet or to Bluetooth for close range communication.

Nodes as well as application instances are identified and addressed using persistent names, the *Endpoint Identifiers (EIDs)*, represented as URIs [11]. They are arbitrary length and combine the equivalent of IP address and port number. The EID concept supports *wildcards* (e.g., prefix matches) so that application instances can register, e.g., for all messages destined to *dtn:http://www.tkk.fi/**. This naming scheme allows for a straightforward mapping of HTTP URIs to DTN addresses and define web-traffic-specific routing rules.

The bundle protocol uses messages—*bundles*—of arbitrary size as basic data unit. Data bundles carry a source and destination EID, numerous fields to control forwarding and reporting, a sender timestamp, a sequence number, an expiry time (equivalent to a time-to-live field but expressed in absolute time, not as hop count), and the payload. Optional extension fields may provide security parameters and metadata, among others. In general, bundle delivery is best effort and, depending on the routing protocol, bundles may be replicated.[4]

Bundles are uniquely identified by their source EID, timestamp, and sequence number; identification may additionally make use of a hash over the bundle content; bundle

[3]Typically, web browsers open multiple parallel TCP connections and make use of HTTP pipelining to speed up the retrieval.

[4]*Custody transfer* may offer some degree of reliability, but its notion of responsible custodians is not well applicable to mobile ad-hoc environments and multi-copy routing protocols so that we do not consider this further.

metadata [31] [24] may provide additional identification information or hints about the content.

The arbitrary size of bundles allows them to be self-contained, i.e., carry either a complete request or response. In particular, a composite web page including all embedded objects may be fit into a single message. For encapsulating HTTP requests and responses, we follow our approach originally described in [20]: we choose MIME encapsulation for HTTP messages that also provides content identification (*message/http*) and allows for aggregating multiple pieces of content via Multipart MIME. The latter is needed for combining all objects embedded in a web page into a single response; we follow the conventions of MHTML [21] and use the *Content-Location* MIME header to identify the individual objects inside the message body.

A DTN-enabled web client issues a single HTTP request encapsulated in a bundle and sends it to a web server. After receiving the request, the server obtains all objects of the requested web page and encapsulates them in a single bundle as described above (using the same time-to-live (TTL) as in the request). We assume that the objects to be included are known from web authoring tools and dynamically generated content on the server side [20]. The server may use end-to-end authentication for bundles or S/MIME to prove authenticity of the contents. When the client receives the response, it extracts the individual objects and renders them according to the *Content-Location* headers. Yet, two issues remain for further study: splitting responses across multiple bundles if they grow too large[5] and dealing with embedded objects that reside on external servers.

### B. Routing, Urban Mobility, and Geographic Awareness

As described above, we have three types of nodes: 1) web clients (= mobile nodes) and 2) web servers (in the fixed network) are distinguished by their EIDs so that the node type of the bundle destination can be identified. 3) Access points in hotspots have the property that they are all interconnected on the fixed network side and thus any access point can serve as a default router for bundles destined to a web server; however, typically the access point forwarding such a request is also a reasonable candidate for delivering the response to the mobile node. We thus split DTN routing conceptually into two parts: 1) routing in the fixed network between access points and DTN web servers and 2) routing in the mobile (ad-hoc) network between mobile nodes and to/from access points.

*1) Fixed Network Routing:* For the fixed network side, we apply deterministic routing. In the simple case (a), the access point forwarding a request establishes a direct connection to the respective web server. The target web server is determined by performing a DNS service or NAPTR record lookup on the domain name encoded in the *hostport* part of the destination URI. The connection is kept open and implicitly a reverse route to the access point is established. In the more complex case of a routing overlay in the fixed network consisting of many bundle routers (including the web server), link-state routing [6] can be used within the overlay to forward the bundles based on the destination EID.

In both cases, an access point receiving a request from a mobile node needs to ensure that the response is routed via itself. No such mechanisms readily exist in the bundle protocol or its extensions. One option would be to create a bundle *return routing* extension block similar to the SIP *Via* header and define a loose source routing mechanism that is then to be employed for routing bundles back to the source EID of the request. Such mechanism would allow the hotspot to insert itself along the path of all web bundles from the server to the client. Alternatively, the access point might change the source EID to include itself as a "path" element [11] so that a response to the source EID would automatically yield the desired result. However, this could invalidate signatures applied to the bundle and eliminate duplicate detection at the server if bundles are forwarded by different access points. A similar duplicate detection problem would occur if we terminated the web client's request in the access point which then generated a new request with its own source EID and created a local mapping between the original and its own request for later forwarding the response.

*2) Mobile Network Routing:* For routing between mobile nodes to and from the access points, we follow the idea of probabilistic routing because it is not possible to keep consistent reachability information (as in mobile IP) in a disconnected environment. Numerous probabilistic *single-copy* and *multi-copy* routing protocols were developed in the past. Out of these, we choose three that are stateless, i.e., do not make forwarding decisions dependent on local per node state such as contact history between nodes. This is important because we want messages to propagate quickly, but stateful probabilistic routing protocols use state primarily to inhibit or direct replication. Also, their operation with the "anycasting"-style delivery via any access point has yet to be explored.

Our variant of *Direct Delivery* [13] only forwards a message directly to the first access point encountered, not relying on other mobile nodes for relaying. With *Spray-and-Wait* [30], the requester (or the access point issuing a response) creates no more than a fixed maximum number of copies and hands them to other mobile nodes for indirect delivery. *Epidemic* routing [32] results in bundles being replicated to every encountered node, thus flooding the network. We also use one variant with a supplementary hop-count limit to restrict the flooding process to the vicinity

[5]While the bundle protocol imposes no size limitations, bundles may need to be passed from one mobile node to the next during a finite time the two nodes are in contact. While fragmentation is supported by the bundle protocol, splitting bundles into many fragments reduces the delivery probability [25].

of the sending node, since we seek quick replies and want to avoid overloading other areas of the network. All three protocols discard bundles when their TTL expires. Access points are configured to accept messages on behalf of the web servers, so that all three probabilistic routing protocols will deliver requests through them. All DTN nodes will detect duplicates based upon the bundle identifiers and discard them, avoiding flooding a server with dozens of identical requests.

*3) Geo-aware Optimizations:* Routing bundles between a mobile client and a fixed server is asymmetric: requests from a mobile node may utilize any of the available access points to reach the server. In the opposite direction, the mobile nodes needs to be located, which is particularly difficult if the request was delivered indirectly via other mobile nodes to the access point since all the nodes including the requester may have moved.

As noted above, as a first approximation, the responses are routed back to the **forwarding access point**. Given sufficiently short processing and transmission time (no more than a few seconds), this will generally cover the case that the mobile node communicates directly with an access point when sending the request. But even for requests that reached the access point indirectly, this seems to be a reasonable approach: if we assume that users will tolerate some but not too much response delay and that they do not move very fast, they could still be found somewhere in the access point's neighborhood.

To extend the reach of responses further and increase the likelihood of indirect response delivery, we introduce the notion of response routing via the **$k$-closest acces points**. The web server returns the response as before to the forwarding access points which then replicates the request to its $k-1$ geographically closest neighbors (e.g., using an online database in which hotspots are registered).[6] Each of the $k$ access points then forwards the response to mobile nodes in reach according to the routing protocol in use. This approach spreads the messages around the area from where the request originated, attempting to cover the entire area where the client may have moved in the meantime.

Both approaches use geographic information to return the response to where the request originated from, but do not require location awareness in mobile nodes, nor do they rely on dynamic exchange of location information, but only use relatively static data.

[6]If geo-aware deterministic DTN routing was available in the DTN overlay in the fixed network, the access point forwarding the request could also insert an address indicating a geographic coverage instead of its own and have the routing performed by the DTN overlay. The decision on the geo range to be covered could be based on whether or not the access point received the request directly from the web client and, if not, how old the request was at the time of forwarding. We leave this for further study.

## C. Caching

For DTN-based web access, we make use of self-contained bundles carrying either the web request or the complete response. Due to the store-carry-and-forward nature of DTN operation, this information may be stored ("queued") in DTN nodes for an extended period of time (until the TTL expires if sufficient storage capacity is available). With multi-copy routing, copies are likely to be spread across several nodes. We have explored such queued or cached contents in mobile nodes to increase the retrieval efficiency for mobile content access in our earlier work [24]: if any node receives a request, it checks—by using EIDs, metadata, or by parsing the MHTML body structure of a message—whether it happens to carry a matching response. If so, it just replicates the response bundle, re-addresses it to the new requester, and sends it as reply.

This approach is feasible for HTTP because many HTTP responses are identical irrespective of the requesting node. To make sure that only suitable replies are generated from caches, the replying node needs to validate that the requester's capabilities (e.g., as indicated in the various *Accept* headers of HTTP) match the content types and encodings of the cached responses. Also, requests should not be handled from caches if cookies are used. For simplicity, we assume homogeneous web clients and web pages without cookies to limit the scope of this paper.

We apply the above optimizations to mobile nodes as well as to access points. The former may yield retrieving web resources faster and/or without any nearby access point. The latter makes perfect caches since they are mains powered, may easily be supplied with extra storage capacity, and naturally serve as aggregation points for requests and may thus help further reduce the burden on the servers. We do not experiment with different caching policies. Mobile nodes only use the routing-protocol-specific queue management, and the access points keep copies until their TTL expires.

## IV. SIMULATIONS

For our evaluation, we use the Opportunistic Networking Environment (ONE) simulator [15]. We run simulations using two different mobility scenarios. First, we use the random way point (RWP) model, which is intuitive and provides a good comparison case. Second, we use the Helsinki City Scenario (HCS), which models pedestrians moving in city streets between interesting locations.

Our scenarios follow 140 pedestrians moving in the Helsinki downtown area ($4.5 \times 3.4$km). In the HCS scenario, the pedestrians walk on streets, and the WLAN access points are located at the streetside. Each pedestrian node represents a user moving with realistic speed along the shortest paths between different points of interest (POIs) and random locations.[7] For comparison, we report on the RWP with the same area and access point locations, but with the pedestrians' movement not limited to streets. Each mobile user in our scenario retrieves a web page at a mean interval of 5 min between requests, uniformly distributed in [280;320]s. Each request chooses a web page randomly from the list of 50 most popular websites requested from within Finland. For each request, we measure the delay and the sender (origin server, hotspot cache, or another mobile node) of the response(s) as well as the number of duplicate requests and responses delivered.

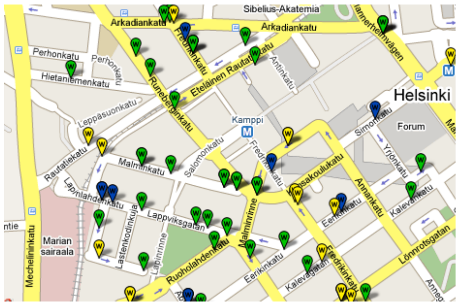

Figure 2. An example communication setting in downtown area.

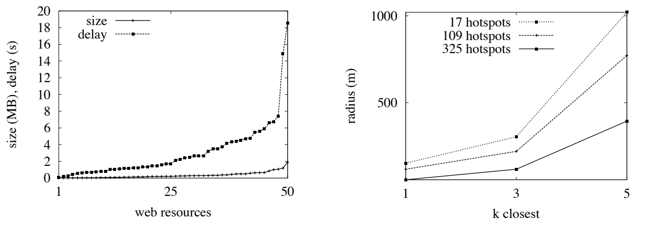

Figure 3. Web resources and average radius covering $k$ APs around an AP.

Figure 2 illustrates a city map section showing the real access point locations in the Helsinki downtown area. We vary the number of WLAN hotspots in the downtown area to observe the effect of infrastructure density. We chose locations of community hotspots that are available to community members.[8]

We run the simulations with different random seeds and varying configurations. Each run lasts for 12 h simulation time and results in over 200k contacts between nodes and 1.5k web access events per datapoint of which we plot the mean (deviations between runs with different random seeds are negligible). The mobile nodes use bi-directional wireless links at 10 Mbit/s data rate and 50 meters communication radius to communicate with each other and with access points (APs). Each node has 512 MB message buffer capacity. The access points run with the same WLAN parameters and provide connectivity to the Internet via non-congested access link.

We model the delay for accessing the web pages and their size distribution based upon measurements carried out in our lab: we ran scripts for retrieving the index pages of the top 50 web sites, recorded the retrieval times, and measured the sizes of the web pages. Figure 3 (left) shows the sizes and delays for all web resources used in our simulations. The delay shows an average of 863 resource retrievals. Figure 3 (right) depicts the mean radius around a hotspot including by the $k$-closest hotspots averaged over all hotspot locations.

While the resource retrieval and caching algorithms operate independently of the routing protocol, the routing protocols often create different numbers of messages in the network and thus affect the number of hotspots and caches reached by a request. For our evaluation, we choose three different routing protocols as noted above: *Direct Delivery* [13], *Binary Spray-and-Wait* [30] with 10 message copies, and *epidemic* routing [32] for flooding without limit on message replication.

In the beginning of simulations all the resources reside in web servers in the fixed network. When a web server is accessed via an access point, the aforementioned measured retrieval delay is introduced in the fixed network. Responses are sent back, but they also start to populate the cache(s) of the respective WLAN access points, so that these can serve subsequent requests directly without the additional retrieval delay. We plot our measurements as a function of message TTL, which limits how long both requests and responses can be forwarded before being discarded (note that the maximum response delay is twice the TTL). We also limit the maximum hopcount to four in our simulations. Figure 6 below shows that this limitation is practical, since the first copy of a response message does not frequently come through a long path—which is to be expected given the short TTL.

Figure 4 illustrates the effect of the message TTL on the probability that a client receives a response; duplicate responses are not counted. As expected, allowing more time to deliver requests and responses significantly increases retrieval success; we also note that even with small TTLs, half of the requests are successful. A comparison between the scenarios shows that, when pedestrians follow city streets, instead of just randomly walking in the area, the retrieval probability increases while other variables stay the same. Expectedly, limiting node freedom increases the chances of passing an access point, leads to longer contacts, and increases contact frequency and thus forwarding opportunities.

We also notice that the difference between the various routing protocols is not very pronounced: direct delivery performs almost as good as flooding, which indicates that direct interaction of mobile nodes with access points dominates

[7]We do not use more realistic models such as the Working Day Movement model [9] since we are interested in the performance of users in the streets.

[8]http://www.wippies.com, for a similar service see, e.g., http://fon.com

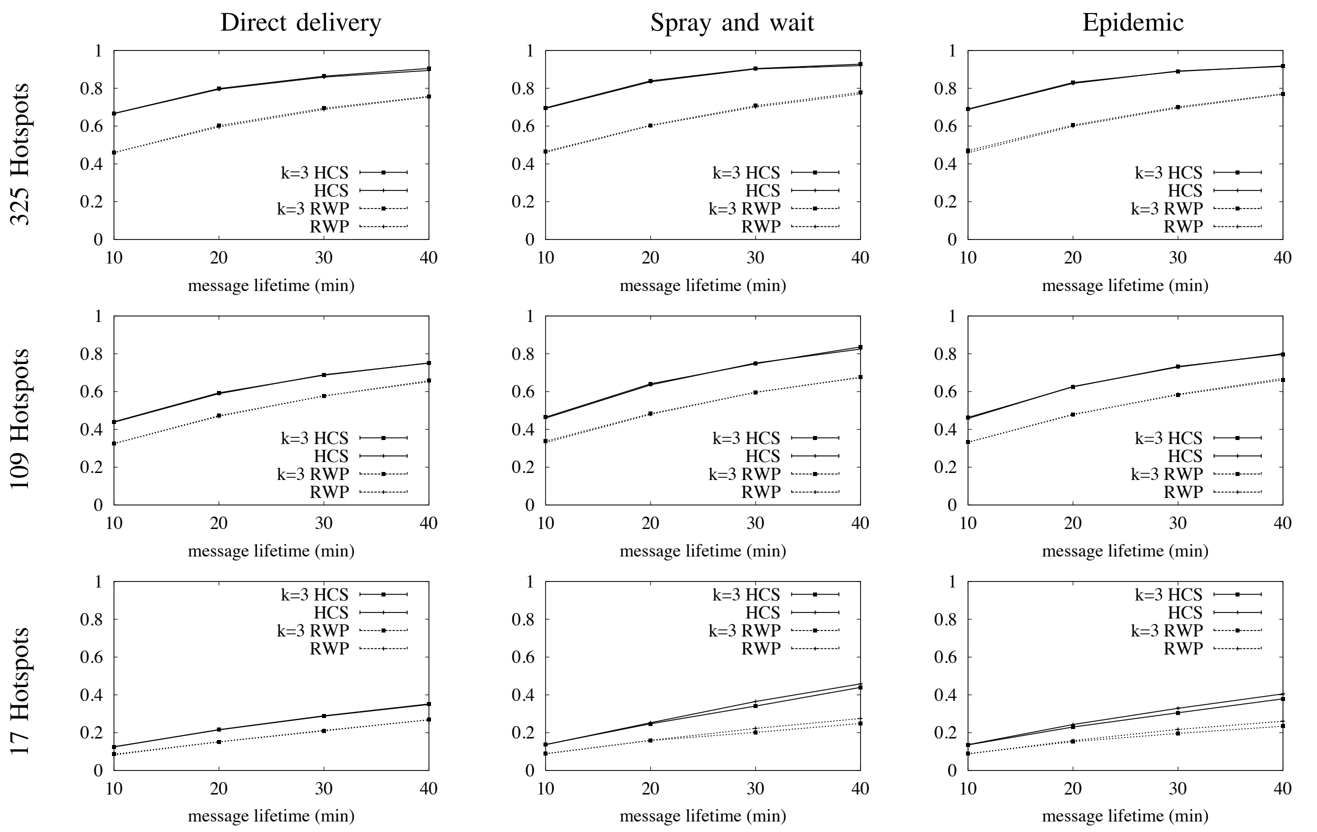

Figure 4. Probability that a pedestrian receives response with different message lifetimes; directly from the closest AP or via $k = 4$ closest APs.

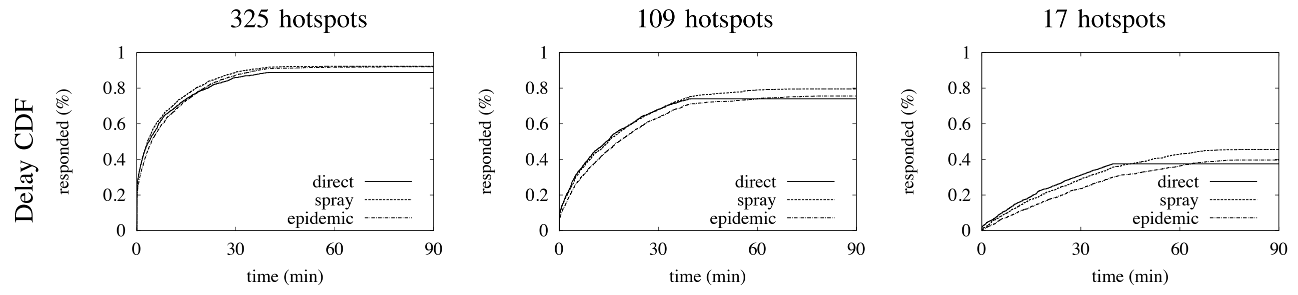

Figure 5. Cumulative distribution function of response delay in HCS scenario with 40 minutes message lifetime.

performance (at least for getting *one* response), probably also due to the small number of nodes.

The density of the available hotspots has a noticeable impact on the retrieval performance. Assuming the HCS model, when using all 325 hotspots and leaving enough time, we get a response ratio $> 0.9$ for the largest and $> 0.75$ for the lowest TTL. Reducing the number of hotspots to about one third (109) still yields ratios of 0.5–0.8 and going down to about 5% (17) hotspots only yields a success ratio of 0.15–0.4. But even this still shows some offloading potential. Extending the reach of responses by applying the $k$-closest replication for responses does not increase delivery probability for nodes that are able to connect to hotspots directly, but the spreading of more message copies is highly beneficial for nodes that rely on other mobile nodes for web access (see below).

Figure 5 shows the cumulative distribution function for response times with TTL=40 min. With a large number of hotspots, nearly 30% of responses are received almost instantaneously. With a smaller number of hotspots, more time passes after request creation before the mobile node reaches a hotspot. A common observation in all scenarios is that even though responses can take as long as twice the message lifetime, most of successful retrievals lead to a first response soon after the request was issued and that waiting longer than roughly TTL does not yield much further increase. This is expected, given that most messages appear to be delivered directly to/from a hotspot: the latter must be reached before the request times out, but then the response will be virtually immediate (which also explain the straight line for Direct Delivery routing after TTL). Then, obviously, the hotspot density has a direct performance impact since it affects the mean time to reach an access point.

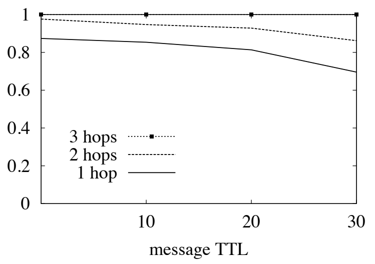

Figure 6. Probability that response has hopcount smaller than n.

Figure 6 shows the response hopcount distribution with Epidemic routing as function of message lifetime with only 17 hotspots deployed and only the closest hotspot responding. We observe that when the message lifetime is large (30 min), only about 70% of responses come from one hop away, whereas with small message lifetimes successful responses arrive always from the closest neighbor. This shows how cooperation between the mobile nodes is important in increasing retrieval success in scenarios with sparse hotspot infrastructure.

Finally, to understand the relevance of indirect hotspot access, we configure 20 pedestrians not to use hotspots directly (i.e., they are not authorized). For these nodes, requests and responses are relayed by other mobile nodes before reaching a hotspot. These nodes can still achieve a response ratio of some 30% with Spray-and-Wait routing, TTL=40 min, and 109 hotspots. Applying the $k$-closest response routing increases this ratio significantly to some 50%, indicating that this controlled redundancy achieves the desired effect. We also observe that caching in mobile clients—which is of little relevance if direct hotspot access dominates—can double the response probability compared just obtaining (cached or forwarded) responses from hotspots. Yet, caching in hotspots can lead to significant 30–70% savings in access link utilization.

Overall, we find that all elements introduced in section III contribute to opportunistic web page retrieval in complementary ways.

## V. Implementation, Validation and Deployment

To validate the practicality of our design we have done a prototype implementation of a complete DTN-enabled web service infrastructure composed of a web server, a hotspot with caching support and mobile web clients. Figure 7 shows a structural overview of the infrastructure elements.

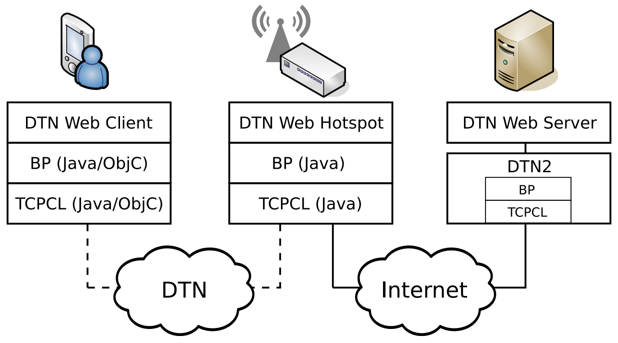

Figure 7. An architectural overview.

The implementations are based on the Delay-Tolerant Network Architecture [4], the Bundle Protocol [27] (BP), and the TCP Convergence Layer Protocol [7] (TCPCL) defined by the Delay-Tolerant Networking Research Group (DTNRG) of the Internet Research Task Force (IRTF). We use the DTNRG reference implementation, DTN2, to provide bundle services to the DTN web server. The hotspot and web client for Android are based on our own Java implementation of the Bundle Protocol and TCPCL, while the iPhone client is based on our own Objective-C implementation of these protocols.

### A. DTN Web Server

The DTN web server is implemented in Ruby and uses the bundle transfer services provided by DTN2. In addition the server includes normal HTTP over TCP service that can be used by standard web browsers. [22]

The DTN web server registers the EID matching the domain name of the web resources that it is serving. For example, a node serving *http://www.dtn.comnet.tkk.fi/dtn/index.html* would register the EID *dtn://www.dtn.comnet.tkk.fi*. The specific resource being sought (*/dtn/index.html*) is carried inside the request message formatted according to RFC 2616.

When the server receives a bundle, it parses the payload for an HTTP GET request. If the payload contains a request for a resource held by the server, it bundles a response message that contains an HTTP 200 OK message and all the objects related to the requested resource using MHTML. These dependencies can be either explicitly listed through a dependency file or automatically generated by parsing the requested HTML file. If the request is invalid, or valid but contains a resource not held by the server, an appropriate HTTP error in returned.

### B. DTN Web Client

We have implemented DTN-capable web clients on two mobile platforms, Android and the iPhone. The clients do not require services from a local bundle node, but instead include their own DTN protocol stacks. These clients allow the user to request web pages, create request bundles, and parse and display the returned content.

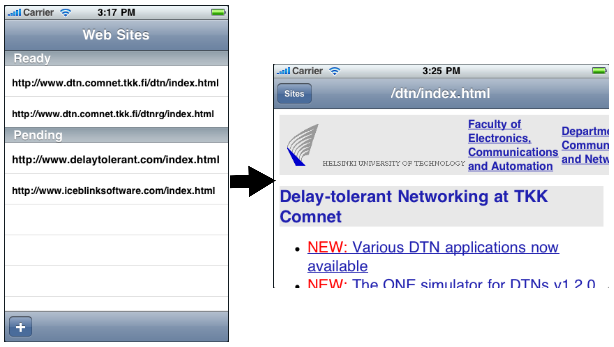
Carrier
3:17 PM
Web Sites
Ready
http://www.dtn.comnet.tkk.fi/dtn/index.html
http://www.dtn.comnet.tkk.fi/dtnrg/index.html
Pending
http://www.delaytolerant.com/index.html
http://www.iceblinksoftware.com/index.html
+
Carrier
3:25 PM
Sites
/dtn/index.html
HELSINKI UNIVERSITY OF TECHNOLOGY
Faculty of Electronics, Communications and Automation
Delay-tolerant Networking at TKK Comnet
NEW: Various DTN applications now available

Figure 8. An example usage view, captured from the iPhone client application.

When a client receives a response bundle from the server, it parses the objects contained within the MHTML message and saves them to the local device storage. When the user wishes to view received websites they are rendered directly from the local file system.

One defining factor of HTML pages is the ability to create links between sites. These links are embedded in the web pages received by the client. There are three ways to deal with linking: first, relatively linked sites on the same web server may be bundled along with the requested site. Second, the client may create request bundles for the linked sites either directly or by user request. Third, the client may fetch the linked sites using the cellular network in case the user selects them. In our current implementation, we opt for the last option in order to not break the users' expectation of immediate feedback after selecting a link.

The workflow of web browsing in well-connected environments typically has the user entering a web site request and actively waiting for the response to arrive. However, response delays over DTN are likely to be significantly longer than the delays users are expecting based on their experience with web browsing in well-connected environments. This makes it infeasible to actively wait for the requested content to arrive. We envision a new web browsing workflow where the user enters multiple sites to be requested, stops using the application while passively waiting for content to arrive, and periodically returns to the application to read the available content. In essence the user expresses interest in certain web resources which are then opportunistically fetched, and transparently updated, in the background. This new workflow requires a user interface that differs from the traditional browsers (only *tabbed browsing* approximates this idea to some extent). We designed our GUI, shown in Figure 8, around a list of requested web sites grouped into two sections, one for the sites that have arrived and another for sites that are still being retrieved. This concept could be extended to include indication for sites whose content has been updated since the previous viewing and to include an option for the user to choose to fetch the site over a cellular data connection instead of DTN.

## *C. DTN Web Hotspot*

We have implemented a DTN web hotspot designed to be run in fixed hotspots and to provide services to the DTN web clients. These services include opportunistic caching, EID to convergence layer address resolution to find DTN web servers, and collusion between multiple hotspots to enable geographic routing. Additionally, a DTN web hotspot could act as a gateway that can respond the DTN web requests by fetching websites from non-DTN web servers and bundling them into response bundles.

The DTN web hotspot includes an opportunistic cache that contains copies of recently requested websites. This allows the hotspot to respond quickly to requests for popular sites directly from the cache saving the resources of its Internet access link.

At the most basic level the hotspot stores bundles that contain web resources in a database and matches incoming request bundles against them. If a match is found the cached bundle is retransmitted by changing the destination EID and the creation time-stamp, leaving the payload untouched. More sophisticated hotspot caching can take advantage of the caching support mechanisms defined by the HTTP/1.1 protocol. These mechanisms allow both the clients and the server to control the caching behavior through headers inserted into the HTTP request and response messages.

Since the original sender of the request bundle may not know the detailed convergence layer addresses required to connect to the web server, the DTN hotspot must be able to resolve the destination EID in the request bundle to the convergence layer addresses. Our implementation uses pre-configured mappings, but since both the hotspot and the server are well-connected to the Internet it would be possible to use the DNS to resolve the EIDs.

## *D. Validation*

We validated our implementations with the scenario depicted in Figure 9. The scenario is composed of an emulated network of buses with DTN2 nodes traveling a 10 km route between Ruoholahti and Otaniemi in the Helsinki city area. Our client implementation connected to one end of the route and the hotspot and web server implementations to another. Our client has been able to retrieve different web sites from the server through the hotspot even though direct end-to-end path between them never existed. This means that the system was able to successfully provide web access to a client 10 km away with no end-to-end connectivity from a hotspot about which the client does not necessarily have any prior knowledge.

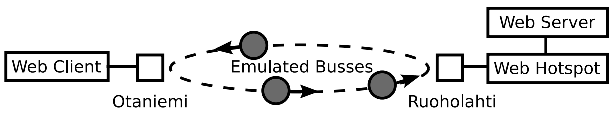

Figure 9. Validation scenario for the implementations.

### E. Deployment

While the implementation described above modifies web clients, web servers and hotspots, in reality modifying all existing network elements in this fashion at once is not feasible. Instead an incremental deployment strategy that operates with current elements is required.

First step of such incremental deployment is to modify those clients and devices that will get immediate benefit from the new functionality. In this case, the whole DTN web functionality can be implemented in the client devices. In this deployment scenario the clients act as DTN web proxies, being able to fetch and bundle web sites from unmodified web servers while connected through unmodified hotspots. Supporting DTN retrieval using a proxy based approach [2], [20] or cooperating without infrastructure support [24] has been discussed in earlier work.

A purely client based solution cannot take advantage of the improved caching and routing opportunities available in cooperating hotspots. However, many commercial and community hotspot networks already deploy an automated access software so the step to add DTN web support is small. The DTN service does not necessarily need to be in the hotspot itself, but can be deployed also as a service in operator's network where it offers TCPCL connectivity through the co-located hotspots. Furthermore, only a subset of all existing hotspots need to provide native support for DTN web.

Finally, while both proxy clients and DTN web enabled hotspots can fetch and bundle web sites from an unmodified web server, bundling done natively at the web server is more efficient and semantically meaningful since the server has the ownership and explicit knowledge about all the related objects that should be bundled into a single reply message. The server can also attach metadata, for example, cache control information, in a centralized manner.

## VI. Conclusions

In this paper, we have investigated one option for reducing the load on cellular networks by offloading selected web interactions of mobile users to a DTN messaging overlay and WLAN hotspots. This feature can also be leveraged to enable access for nodes without cellular connectivity. The presented approach seems suitable for applications that can tolerate some retrieval delay.

We find that the density of commercial and community hotspots in a real-world urban setting is sufficient to satisfy the majority of the requests issued by pedestrians in a short period of time ($\leq$ 20min), 90% being satisfied within little more than one hour. The ratio can be moderately improved further if several adjacent hot-spots are involved in spreading the responses. While the number of hotspots available is—expectedly—key to the performance, cooperation between mobile nodes using DTN ad-hoc routing surprisingly only yields minor improvements for the users that are able to directly access the hotspots.

The positive impact of cooperative messaging among mobile nodes is visible only in scenarios with lower access point density and higher acceptable delay. Moreover, we observe that caching in mobile nodes and hotspots can improve the retrieval ratio and reduce the delay further in all setups, and is especially important for the mobile nodes that cannot directly access the hotspot infrastructure. These nodes also gain the greatest benefit from controlled replication of the response messages. Applying caches in hotspots can significantly help to reduce messaging between hotspots and the fixed network.

Our prototype implementation shows that these concepts can be put into practice, but also that further work on how applications should use delay tolerant communications is needed. Moreover, new user interface designs are needed for making DTN-based mobile web access intuitive to users.

Our simulation-based evaluation work has been constrained to one scenario. In our future work, we seek to explore a variable number of users with diverse user capabilities and cover faster moving mobile users (e.g., in buses). We also want to investigate more diverse load scenarios, especially to understand if the short-lived messages and the bounded spreading can ensure operation of the system across a broader scale. We are also interested in dynamically adapting the geographic spraying, for example, as a function of the age of a request. In addition, applicability of the communication model for information access in very sparse scenarios [18] continues to be an interesting topic. Finally, we are expanding our prototype to become part of the DTN-Bone and provide practical web access from mobile devices via real hotspots.

## Acknowledgments

This work was partly funded by the Academy of Finland in the DISTANCE project (grant no. 117429) and by Teknologiateollisuus ry in the REDI project.

The authors would like to thank Lauri Peltola for the implementation of the DTN web server and client proxy as well as Yunqiu Zhong for the website data collection.

## References

[1] M. Afanasyev, D. G. Andersen, and A. C. Snoeren. Efficiency through eavesdropping: link-layer packet caching. In *NSDI'08: Proceedings of the 5th USENIX Symposium on Networked Systems Design and Implementation*, 2008.

[2] A. Balasubramanian, B. N. Levine, and A. Venkataramani. Enhancing interactive web applications in hybrid networks. In *Proceedings of the 14th ACM international conference on Mobile computing and networking (MobiCom)*, 2008.

[3] M. Bechler, W. J. Franz, and L. Wolf. Mobile Internet Access in FleetNet. In *13. Fachtagung Kommunikation in verteilten Systemen, Leipzig, Germany*, April 2003.

[4] V. Cerf, S. Burleigh, A. Hooke, L. Torgerson, R. Durst, K. Scott, K. Fall, and H.Weiss. Delay-Tolerant Network Architecture. RFC4838, 2007.

[5] CHIANTI Project. Operational and User Requirements. Deliverable D1.2, June 2008.

[6] M. Demmer and K. Fall. DTLSR: Delay Tolerant Routing for Developing Regions. In *ACM SIGCOMM Workshop on Networked Systems in Developing Regions*, Aug. 2007.

[7] M. Demmer and J. Ott. Delay Tolerant Networking TCP Convergence Layer Protocol. Internet Draft draft-irtf-dtnrg-tcp-clayer-02.txt, Work in Progress, November 2008.

[8] F. R. Dogar, A. Phanishayee, H. Pucha, O. Ruwase, and D. G. Andersen. Ditto: a system for opportunistic caching in multi-hop wireless networks. In *Proc. ACM MobiCom*, 2008.

[9] F. Ekman, A. Keranen, J. Karvo, and J. Ott. Working day movement model. In *Proc. of the 1st SIGMOBILE Workshop on Mobility Models for Networking Research*, May 2008.

[10] K. Fall. A Delay-Tolerant Network Architecture for Challenged Internets. Proc. of ACM SIGCOMM 2003, Computer Communications Review, Vol 33, No 4, August 2003.

[11] K. Fall, S. Burleigh, A. Doria, J. Ott, and D. Young. The DTN URI Scheme. Internet Draft draft-irtf-dtnrg-dtn-uri-scheme-00, Work in progress, March 2009.

[12] J. Greifenberg and D. Kutscher. Efficient publish/subscribe-based multicast for opportunistic networking with self-organized resource utilization. In *AINAW '08: Proceedings of the 22nd International Conference on Advanced Information Networking and Applications - Workshops*, 2008.

[13] S. Jain, K. Fall, and R. Patra. Routing in Delay Tolerant Networks. Proceedings of the ACM SIGCOMM, 2004.

[14] G. Karlsson, V. Lenders, and M. May. Delay-tolerant Broadcasting. In *ACM SIGCOMM Workshop on Challenged Networks (CHANTS)*, 2006.

[15] A. Keränen, J. Ott, and T. Kärkkäinen. The ONE Simulator for DTN Protocol Evaluation. In *Proceedings of the 2nd International Conference on Simulation Tools and Techniques (SIMUtools)*, 2009.

[16] J. Leguay, A. Lindgren, J. Scott, T. Friedman, and J. Crowcroft. Opportunistic content distribution in an urban setting. In *ACM SIGCOMM workshop on Challenged networks (CHANTS)*, pages 205–212, 2006.

[17] S. Lim, W.-C. Lee, G. Cao, and C. R. Das. A novel caching scheme for improving internet-based mobile ad hoc networks performance. *Ad Hoc Networks*, 4(2):225 – 239, 2006.

[18] T. K. M. P. J. O. Md. Tarikul Islam, Anssi Turkulainen. Practical voice communications in challenged networks. In *At the 1st Extreme Workshop on Communications*.

[19] E. Nordström, P. Gunningberg, and C. Rohner. A search-based network architecture for mobile devices. Technical Report 2009-003, Uppsala University, 2009.

[20] J. Ott and D. Kutscher. Bundling the Web: HTTP over DTN. In *Proceedings of WNEPT 2006*, August 2006.

[21] J. Palme, A. Hopmann, and N. Shelness. MIME Encapsulation of Aggregate Documents, such as HTML (MHTML). RFC 2557, March 1999.

[22] L. Peltola. Enabling DTN-based Web Access: the Server Side. Master Thesis, Helsinki University of Technology, Department of Communications and Networking, 2008.

[23] M. Pitkänen, T. Kärkkäinen, J. Greifenberg, and J. Ott. Searching for Content in Mobile DTNs. In *7th Annual IEEE International Conference on Pervasive Computing and Communications PerCom*, March 2009.

[24] M. Pitkänen and J. Ott. Enabling opportunistic storage for mobile dtns. *Journal on Pervasive and Mobile Computing*, 4(5):579–594, 2008.

[25] M. J. Pitkänen, A. Keränen, and J. Ott. Message Fragmentation in Opportunistic DTNs. In *2nd WoWMoM Workshop on Autonomic and Opportunistic Communications*, June 2008.

[26] J. Scott, P. Hui, J. Crowcroft, and C. Diot. Haggle: A Networking Architecture Designed Around Mobile Users. In *Proceedings of IFIP WONS*, January 2006.

[27] K. L. Scott and S. C. Burleigh. Bundle Protocol Specification. RFC5050, Nov. 2007.

[28] A. Seth, P. Darragh, S. Liang, Y. Lin, and S. Keshav. An architecture for tetherless communication. In M. Brunner, L. Eggert, K. Fall, J. Ott, and L. Wolf, editors, *Disruption Tolerant Networking*, number 05142 in Dagstuhl Seminar Proceedings, 2005.

[29] G. Sollazzo, M. Musolesi, and C. Mascolo. Taco-dtn: a time-aware content-based dissemination system for delay tolerant networks. In *MobiOpp '07: Proceedings of the 1st international MobiSys workshop on Mobile opportunistic networking*, pages 83–90, 2007.

[30] T. Spyropoulos, K. Psounis, and C. S. Raghavendra. Spray and wait: an efficient routing scheme for intermittently connected mobile networks. In *WDTN '05: Proceeding of the 2005 ACM SIGCOMM workshop on Delay-tolerant networking*, pages 252–259, 2005.

[31] S. Symington. Delay-Tolerant Networking Metadata Extension Block. Internet Draft draft-irtf-dtnrg-bundle-metadata-block-00, Work in progress, September 2008.

[32] A. Vahdat and D. Becker. Epidemic routing for partially connected ad hoc networks. Technical Report CS-200006, Duke University, April 2000.

[33] L. Yin and G. Cao. Supporting cooperative caching in ad hoc networks. *IEEE Transactions on Mobile Computing*, 5(1):77–89, 2006.