Information Delivery in Tetherless Healthcare

PJ Dillon
University of Pittsburgh
210 S. Bouquet St, Rm 6135
Pittsburgh, PA 15260
pdillon@cs.pitt.edu

Taieb Znati
University of Pittsburgh
210 S. Bouquet St, Rm 6135
Pittsburgh, PA 15260
znati@cs.pitt.edu

ABSTRACT

Tetherless care was proposed to help address the costly burden of chronic conditions and diseases like diabetes, hypertension, and heart disease. In support of this vision, this work presents a solution for the intelligent delivery of real-time messages given intermittent connectivity and limited energy. It employs a O(1) Markov predictor operating over a history of network sessions to predict likely future low power opportunities to transfer data while attending to real-time delivery deadlines. The algorithm was deployed to the smartphones of several volunteers for two months and was tasked with managing the transfer of test data and statistics of its operation. Results show:

- Predictions of the duration or start time of a given session have 80% accuracy to within six minutes.
- Where delays between successive WiFi sessions are less than nine minutes with 81% probability, the system is capable of supporting deadlines on the order of minutes with WiFi-based sessions only, falling back on more costly cellular technology to cover the final 19% of delays.
- With a fixed 24 hour deadline for all messages, the solution can often introduce further delay to conserve energy, waiting for the advent of some future session before initiating transmission.

General Terms

tetherless care, tetherless patient

Keywords

mobile healthcare, ubiquitous computing, delay tolerant, intermittent connectivity, tetherless care

BODYNETS 2013, September 30-October 02, Boston, United States

DOI 10.4108/icst.bodynets.2013.253698

1. INTRODUCTION

Healthcare systems in the industrialized world are plagued with a wide array of issues, but among them are an aging population stricken with high rates of costly yet preventable diseases, such as diabetes, hypertension, and heart disease. Poor lifestyle choices, inefficiencies in current delivery practices, and prolonged hospital stays requiring continuous monitoring contribute to the costs of care. Yet emerging technologies in sensing and wireless communication are providing new opportunities to address these issues and change the way healthcare is delivered. While some diseases and chronic conditions require patient hospitalization, a large and potentially growing class of health conditions and sicknesses could easily and more cost-effectively be treated in an outpatient manner, all without compromising the quality of care.

Traditional in-hospital care allows the continuous monitoring of vital signs, real-time and reliable reporting of alerts of abnormal conditions, and prompt feedback from and interaction with caregivers.

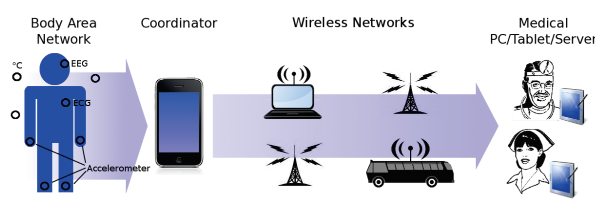

Figure 1: The Tetherless Patient

To provide comparable care, we proposed the concept of the *tetherless patient* [5], an unencumbered mobile patient monitored remotely through a system of sensors and wireless networks, as illustrated in figure 1. To support this concept, an architecture was proposed that facilitates the adaptation of system operation to the available power, sensing, and computing infrastructure such that the patient's safety is never compromised. A critical aspect of this architecture, which is the focus of this paper, is its ability to achieve reliable, real-time, and pervasive communication between the patient and the care giver.

The architecture assumes the existence of a system of sensors, organized into a low-power wireless network and placed on or around the patient, that gathers data on the patient's smartphone. The main components of the architecture running on the smartphone support a set of tetherless care applications tasked with (1) monitoring vital signs and environmental signals, (2) analyzing and reacting to the patient's state, and (3) alerting appropriate caregivers to important physiological changes in a context-aware manner. The communication between the smartphone and the caregivers can be achieved using (1) low-power opportunistically available wireless networks or (2) more costly cellular network technology. To enable greater patient mobility, the architecture must ensure continued interaction between the patient and caregivers. This requires availability of wireless connectivity and careful management of critical power resources to enable information exchange. In a dynamically changing environment, however, the availability of these resources may not be guaranteed.

To ensure patient safety, mechanisms must be in place to predict the patient's itinerary, based on learned behavior, and project connectivity opportunities and power expenditures along the patient's path. Based on the projected connectivity and the condition of the patient, a decision must be made regarding when to initiate information exchange and which opportunities along the path are selected to do so.

This paper proposes algorithms for patient itinerary prediction and communication scheduling to enable the context-aware, real-time delivery of information between the patient's smartphone and the caregivers. More specifically, the main contributions of the paper are:

1. An algorithm, based on a O(1) Markov predictor, for projecting network connectivity as a set of successive sessions between the patient's smartphone and the least costly wireless communication networks along the patient's path.

2. An algorithm for intelligently delaying real-time message transmission to achieve energy savings without compromising the real time data requirements.

3. A proof-of-concept implementation of the tetherless care system on a set of Android-based devices.

The remainder of this paper is organized as follows: Section 2 discusses prior work towards the tetherless care vision; Section 3 presents three algorithms for processing network session logs, predicting future sessions, and deciding when to upload information; Section 4 outlines the methodology for deploying and investigating the algorithm's performance; Section 5 presents results obtained from a two-month trace of Android-based devices carried by volunteers; Section 6 enumerates several new research paths; and Section 7 concludes the work.

2. RELATED WORK

A number of applications and systems have been proposed towards enabling tetherless care: a survey of these applications can be found in [14]. The majority of these applications are based on a networked set of sensors placed on the patient. Some introduce a central, more powerful node, such as a smartphone to collect and process sensor data[7, 17, 19, 18, 15, 21, 10]; others rely on off-body computing[16, 11, 20, 13]. These works do not address the challenges introduced by full patient mobility, opting instead to either explore confined scenarios or assume the existence of a ubiquitous well provisioned wireless network, such as WiFi or cellular technology. The perspective taken by tetherless care is that the system must not rely solely on one network or communication technology but instead exploit any and all communication methods in a safe, secure, and low-power manner.

Balasubramanian et al. [2] studied mobile network availability in urban settings, finding 3G networks to have 87% availability and WiFi 11% availability. Furthermore, the study shows that 3G provided a consistent and sustained throughput but at generally higher energy cost. On the other hand, WiFi, under good conditions, provided greater throughput at lower energy cost. Under poor conditions, WiFi may require more energy due to retransmissions. Huang et al. [9] found that 4G/LTE networks, in general, provide greater throughput than WiFi but with an energy cost exceeding both 3G and WiFi. As such, the tetherless care framework must intelligently utilize these available technologies to meet the demands of its applications in the most energy efficient manner.

The problem of intermittent connectivity has been considered more generally in the Delay Tolerant Networking (DTN) community [8, 12, 1], which supports the transfer of data across multiple intermittently available links over a vast range of timescales. The work described in this paper differs insomuch that it introduces real-timeliness and assumes the intermittent connections only occur between the patient's device and various access points. To the best of the authors' knowledge, this is the first work to consider real-timeliness in the face of intermittent connectivity.

3. INFORMATION DELIVERY IN TETHERLESS CARE

With our assumption of intermittent network availability, we can model the user's day-to-day networking as a time-ordered series of network sessions. Each session is a tuple, $\langle id, type, t, d, r, v \rangle$, where id is an identifier of the remote entity to which a connection is made; $type$ identifies the underlying technology of the session, e.g. 802.11g/n, EVDO, LTE, bluetooth; t is the session start timestamp; d is the duration of the session; r is the measured throughput achieved during the session; and v is an optional location or venue identifier. Where the focus of this work is on 802.11g/n, id is set to the MAC address of the access point to which the device connects.

Data to be transferred is grouped into messages where each message has an identifier, payload, size, and deadline. The objective of the information delivery component is, for each message, to find a likely set of future sessions, if more than one is needed, capable of delivering the message before its deadline while minimizing the power consumed.

To solve this problem, the system employs an $O(1)$ Markov predictor operating over the series of network sessions to construct a tree of possible and likely future sessions where the current session is at the root of the tree and each child represents one possible session likely to follow its parent. The algorithm then schedules a given set of messages along each path in the tree starting at the leaves using an earliest-deadline-first approach. If each path offers sufficient bandwidth to upload the entire load in subsequent sessions while meeting each message deadline, the system defers transfer to some future time. Otherwise, a transfer is started in the current session. For power consumption purposes, the algorithm constructs separate schedule trees for Bluetooth, WiFi, and WWAN (3G/4G), favoring Bluetooth or WiFi before WWAN and falling back only when one tree is overloaded.[1]

The algorithm is broken into three parts. Subsection 3.1 describes a mechanism to analyze session history and determine the temporal adjacency between current and subsequent sessions. Subsection 3.2 describes the session prediction algorithm to determine the duration of a given current session, the start time of a subsequent session, and the construction of the tree of likely future sessions. Subsection 3.3 describes an algorithm for scheduling messages along each branch in the tree to ultimately determine if data transfer can be deferred to a future session, thereby saving power. Finally, subsection 3.4 describes how this computation potentially contributes to application-dependent risk analysis.

3.1 Session Correlation Algorithm

The objective of this algorithm is to construct a two-dimensional mapping from indices of some current session, s, and a potential subsequent session, b, to a list of associated metadata describing the transition from s to b. For the current session, s, the time-ordered series of m prior sessions, $L = s_{t-m+1} \dots s_{t-1}s_t$, is searched for all prior instances of s and the subsequent session, b, of the same *type* as s. In the map, M, at location $M[s.id][b.id]$, the tuple of $\langle s.t, s.d, b.t - s.t\rangle$ is added to a list ordered by $s.t$.

Algorithm 1 Log analysis algorithm

```
Require: network session s, session log L, map M
  L' ← findAllByID(L, s.id)
  for all sessions i in L' do
    b ← findSubsequentSessionOfType(s, s.type)
    push(M[s.id][b.id], ⟨s.t, s.d, b.t − s.t⟩)
  end for
```

This mapping can be easily implemented using cloud storage technologies, such as BigTable [3] or Cassandra. The row key is a concatenation of a user identifier, the current network session identifier, *s.id*, and any venue identifier in *s.v*. A column family would be defined for the next sessions where each column would be identified by *b.id*. The start timestamp of the current session would be used as the timestamp for the table entry, which would store the tuple, and the versioning history of the storage would compose the list.

3.2 Session Prediction

The prediction algorithm estimates the duration of the given current session and the start time of the following session using a Markov predictor based on a context of prior sessions. Given a time series of sessions, $L = s_{t-m+1} \dots s_{t-1}s_t$, the probability of arriving at a future session, b, is expressed as:

$$P(s_t, b) = P(X_{t+1} = b \mid X_t = s_t) = \frac{N(X_t b, L)}{N(X_t, L)},$$

where X_t represents the state of the system at time t, $N(q, L)$ denotes the number of times the sequence q occurs in the context L. Given a map, M, the numerator is the number of entries stored in the list at $M[s_t.id][b.id]$, and the denominator is the total number of entries in each list across the row at $M[s_t.id]$. The expected duration of s_t and the expected start time of b are computed from the respective prior values stored in each tuple of the list at $M[s_t.id][b]$.

[1]A solution involving one, integrated tree that expresses the energy and monetary cost of each session and subsequently mitigates their use is left for future work.

Algorithm 2 Session Prediction

```
Require: root tree node r, current session s_t, map M, min
  probability Φ
  SCHEDULE((r, s_t, M, 1.0, Φ))
  function SCHEDULE(n, s_t, M, p_s, Φ)
    total ← 0
    for all list ℓ ∈ M[s_t.id] do
      total ← total + length(ℓ)
    end for
    for all ⟨session b, list ℓ⟩ ∈ M[s_t.id] do
      p_b ← length(ℓ) ÷ total · p_s
      if p_b > Φ then
        dur ← avgDuration(ℓ)
        t ← avgNextStartTime(ℓ)
        c ← TreeNode()
        c.edge ← p_b
        c.parentDuration ← dur
        c.expectedStart ← s_t.t + t
        addChild(n, c)
        SCHEDULE((c, b, M, p_b, Φ))
      end if
    end for
  end function
```

Algorithm 2 recursively constructs a tree representing all likely future sessions, where each node corresponds to a given session and where each edge is marked with the probability of arriving at the child session after the parent. For a given session, s_t, and for each potential future session, b, at the row $M[s_t.id]$, the algorithm computes the probability, p_b, of initiating b after s_t. If p_b is greater than some threshold probability, Φ, it adds a child node to the tree that stores the expected duration of the parent node for session s_t and the expected time at which b will start.

The algorithm is repeated for each available network technology such that the system can utilize the constructed tree to select the lowest power technology first; falling back to the others only when needed.

3.3 Message Scheduling

A message is defined to be a tuple of $\langle id, payload, size, deadline\rangle$, and the message queue is assumed to be sorted by ascending deadline. Based on the session prediction tree, the information delivery component determines if enough bandwidth is expected in future sessions to delay the delivery of a given message load without violating the delivery deadline of any message. Note that the root node of the schedule will always exist but may have an expected duration of zero if no prediction can be made about the current session, i.e. if this is the first encounter with the associated access point.

In the base case, corresponding to a leaf node and its respective session, the algorithm allocates a slot for each message along the time window defined by the session's duration. The slot size is determined by the message transfer time, computed as the ratio of the message size and the session's observed average data rate. Furthermore, slots are time ordered based on their deadline, and messages are served using the Earliest Deadline First scheduling algorithm. The algorithm then returns a tuple, $\langle wait, miss,$

Algorithm 3 Message Scheduling

```
Require: schedule tree r, message queue Q
  tuple u ← canWaitForNextSession(r, Q)
  if not u.wait or u.miss then
      StartUpload( )
  end if
  function canWaitForNextSession(n, Q)
      s ← n.s
      if size(children(n)) = 0 then
          w ← s.d × s.r
          ⟨d, m⟩ ← scheduleMessages(s.t, s.t + s.d, Q, s.r)
          return ⟨false, m, w − d, 0⟩
      end if
      tuple min ← ⟨false, true, ∞, 0⟩
      for all node c ∈ children(n) do
          s ← c.s
          w ← s.d × s.r
          ⟨d, m⟩ ← scheduleMessages(s.t, s.t + s.d, Q, s.r)
          u ← canWaitForNextSession(c, Q)
          if w + u.here < min.here then
              min.wait ← u.here > 0 and not u.miss
              min.miss ← m
              min.here ← w + u.here
              min.next ← u.here
          end if
      end for
      return min
  end function
  function scheduleMessages(t_s, t_e, Q, r)
      miss ← false
      for all message m ∈ Q do
          t_m ← m.size ÷ r
          if t_s + t_m > t_e or m.deadline < t_s + t_m then
              miss ← true
          end if
          t_s ← t_s + transferTime
      end for
      return ⟨t_e − t_s, miss⟩
  end function
```

$here, next\rangle$, where $wait$ is a boolean set true if all future sessions can service the queue, which is always false in the base case; $miss$ is a boolean set true if one or more deadlines would be missed during this session; $here$ is the number of free, unscheduled milliseconds available in this session, which can be negative; and $next$ is the minimum number of free, unscheduled milliseconds available in some future session, which is zero for the base case and can be negative.

In the recursive case, for some non-leaf node and its respective session's window size, w, in milliseconds, the algorithm first schedules each message in the session's window to determine if any deadline would be missed, setting its $miss$ return value accordingly. It then collects the tuples from each of its children and selects the tuple, u, where the sum of $u.here$ and w is a minimum. Finally, it returns a similar tuple where $wait$ is true if $u.here > 0$, $next$ is set to $u.here$, and $here$ is set to the sum of $u.here$ and w, i.e. prepending the opportunity w provides to the schedule computed in the future session.

The final resulting $wait$ and $miss$ values determine if transmission can safely be delayed to save power, which must be set true and false, respectively.

This algorithm selects the path through the schedule with the minimum bandwidth to account for the worst case. If all possible paths provide future opportunities for upload, the system can delay transmission with reasonable certainty. Otherwise, for safety, energy should be expended to upload data during the current session.

3.4 Risk

A freely roaming patient introduces an element of risk into his or her context, for which tetherless care applications must be able to account. To assist with this endeavor, the system provides a measure of the risk of the patient's context.

Risk is defined here as the ability of the system to meet some potential future deadline. The highest risk situations occur when a schedule is unknown, when established deadlines have already elapsed, or when insufficient bandwidth is expected to service an existing queue. It grows proportional to the amount of data to upload and inversely proportional to the expected available bandwidth. And the lowest risk situations occur when unoccupied bandwidth is available or expected to be available in the near future.

As such, the final $here$ and $miss$ values computed in algorithm 3 serve as a proxy measure for risk. Large values of $here$ indicate low risk situations. As the message load increases, its value decreases. And, when its value is negative, the system cannot guarantee on-time delivery.

This measure is intended neither to be an all-inclusive measure of risk nor capable of modeling sudden changes in the patient's risk. Instead, the intention is to provide a complement to applications assessing and responding to the patient's risk in an application dependent manner.

4. METHODOLOGY

The algorithm described above is designed to execute in both a cloud computing environment and on a mobile device. The mobile device must be capable of executing the algorithm in a stand-alone fashion in a limited capacity when cloud computing services are not available for network availability or power reasons. To assess the feasibility of the proposed message delivery infrastructure, a proof-of-concept implementation of each of the algorithms was developed to support the tetherless care architecture running exclusively on the mobile device. Without loss of generality, every session was assumed to have a bandwidth capacity of $400kbps$, or 50 bytes per millisecond, and the parameter Φ was set to 0.2. Also, the implementation was confined to monitor, predict, and utilize WiFi-based sessions to exploit their intermittent availability and prevent impacting the cellular data usage of volunteers in the study.

A tetherless care application was developed to generate messages for the system containing the results of instrumenting the implementation. Also, the application implemented a location monitoring algorithm based on three sources of location information from the Android OS [4] and generated messages regarding the devices location over time. In all cases, message deadlines were set to a fixed 24 hours.

The architecture and application were deployed for a Motorola Droid Pro running Android 2.2 from Verizon with 1GB of internal storage and a Samsung Galaxy Player running Android 2.3.6 with 5GB of internal storage, both of which were carried by the author for a period of two months. In addition, volunteers were petitioned to install the software on their personal Android devices of unknown and varying capabilities.

Android 2.2, unlike more recent versions, does not initiate WiFi sessions while the screen is off, instead requiring user interaction with the device before one is initiated. Where the author carried it as an ancillary device, it was less able to capture and monitor his circadian rhythms.

A log of the WiFi network sessions was collected for a period of up to four months and stored on each device. The first two months, or less, were designated a warmup learning period and, along with the continued learning of the latter two months, used as the basis for the algorithms' computation. As each session ended, a comparison was made between its duration and, if a schedule existed, the average durations computed from all prior instances of the session and stored in each egressing edge of the schedule, i.e. each outgoing edge of the schedule computes its own expectation of the current session's duration to which the measured duration is compared. Similarly, as the next session starts, the computed average and median start times for prior instances of the new session were compared against its real start time. The number of times the system delayed a transmission until a future session was counted along with the remaining available bandwidth (the risk), the number of missed deadlines, and the actual measured speed of each transmission.

5. RESULTS

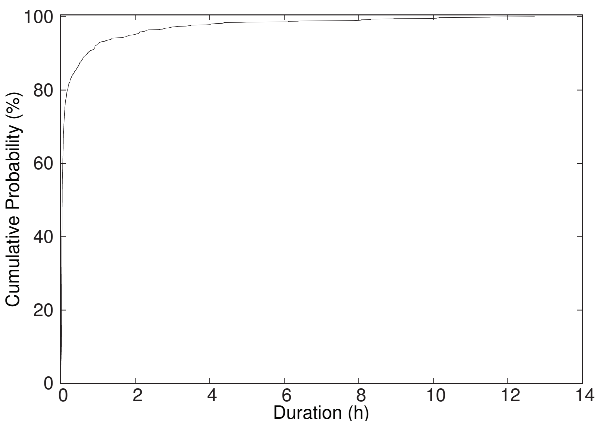

Figure 2: Duration Distribution

Over the two months of the experiments, volunteers encountered 16 WiFi access points, visiting 9 of them repeatedly, and initiated a total of 1442 sessions. Figure 2 plots the cumulative probability of the duration of all the collected sessions. Durations range from several seconds to several hours with an average of 0.36 hours (21.6 minutes) and a sample standard deviation of 1.2 hours.

Figure 3 plots the cumulative probability of the difference between each measured session duration and the most accurate predicted duration of the egress edges in the session's schedule, which exhibit an overall average of 0.29 hours and a sample standard deviation of 0.88 hours. Furthermore, the difference correlates to the length of the duration with a coefficient of 0.70; that is, as the duration increases, the prediction tends to lose accuracy. The figure shows that 80%

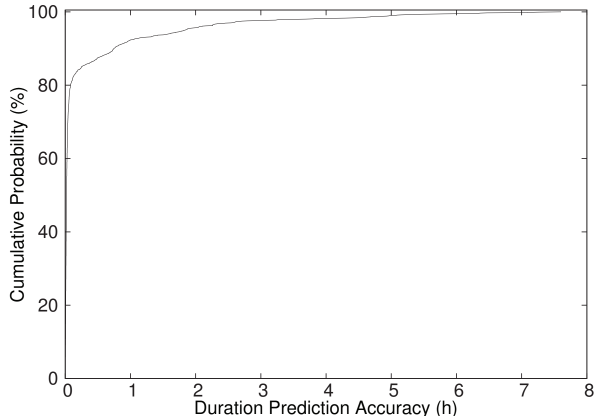

Figure 3: Accuracy of Duration Prediction

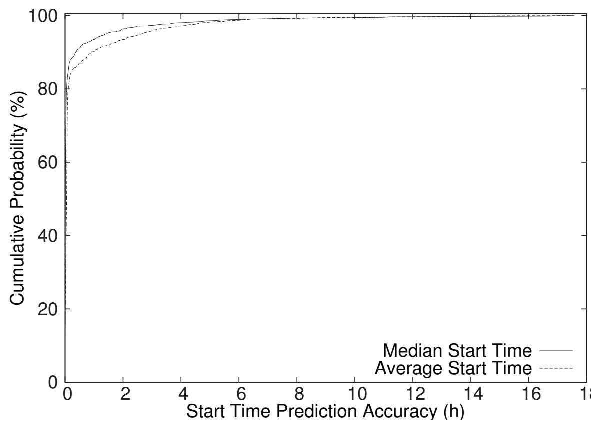

Figure 4: Accuracy of Session Start Prediction

of the predictions are within six minutes.

Figure 4 plots the cumulative probability of the difference between the predicted start time and the actual measured start time of a given session. The difference of the average from the measured exhibits an average value, itself, of 0.42 hours and sample standard deviation of 1.41 hours whereas the difference of the median and the measured exhibits an average of 0.30 hours and a sample standard deviation of 1.35 hours. Once again, the figure demonstrates that 80% of the predictions are within six minutes for the average start time and three minutes for the median start time.

Together, figures 3 and 4 demonstrate that, with high probability, the predicted schedule matches the observed schedule but introduces error into the bandwidth estimation used when scheduling. For this study, the average amount of data uploaded during each session was $58.5kB$ with a sample standard deviation of $279.3kB$, and thus it was observed that most transfers consumed only a few milliseconds in a given session such that the needed transfer time was likely well within the margin of error of the estimated bandwidth. This raises questions regarding the bandwidth demands of a more representative tetherless care application, whether any

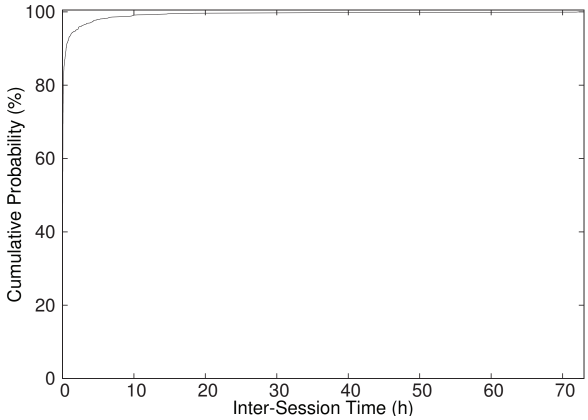

Figure 5: Inter-Session Interval Distribution

tetherless care application would approach the limit of the estimation, and what that limit is. These are left for future work.

Figure 5 plots the inter-session time distribution; that is the distribution of the time interval between the end of one session and the start of its subsequent session, which is a common measure in delay-tolerant networks since it illustrates the largest component of the delay experienced by data traversing the network. This chart indicates the deadline of 24 hours for this study was lax enough to encompass 99.6% of the delays experienced. Also, 81.2% of delays were less than nine minutes, suggesting that the intermittently connected WiFi environment is capable of supporting minimum deadlines on the order of minutes, and other network technologies are likely necessary to circumvent the last 18.8% of the delays introduced by the disconnected WiFi radio. Nevertheless, a significant opportunity exists to minimize the costs of utilizing more expensive networks.

The message scheduling algorithm was often executed more than once per session if the device remained stationary such that it could determine the venue the user was currently visiting, which seemed to have no discernible effect of the algorithms for WiFi-based sessions but would be necessary to differentiate WWAN-based sessions. The average depth of the schedule trees created was 1.08 with a max of 4. With Φ set to 0.2, schedules had a max branching factor of 5. Given the relatively light load of this study, none of the schedules were ever predicted to be overloaded. Also, 80.6% the scheduling executions determined that transfers could be delayed. For a more complete analysis of the scheduling algorithm, the reader is referred to the technical report in [6].

6. FUTURE WORK

Still to explore is the relationship between the mobile device and cloud computing services. The four months of the study produced a $138kB$ log sqlite database, suggesting that contemporary devices have ample storage to accommodate the session history. However, iterating over that history, while not measured, was visibly time consuming. As the log grows, better performance may be achieved by storing the entire log on the server and maintaining a truncated version on the device and a subset, if needed, of the map M described in Section 3.1. The device and server could engage a protocol to (1) entirely compute the schedule on the server and deliver it to the device when possible and (2) synchronize the mobile storage such that the device can compute a schedule independently in poorly connected environments. Additionally, this could potentially allow the server to exploit knowledge collected by multiple users in the system. More work is needed to understand the underlying fundamental question of how little history is needed to achieve and maintain good prediction performance and what gains the server can offer above that.

7. CONCLUSION

This paper presented a set of algorithms for real-time information delivery given limited network availability and limited power in support of the tetherless patient vision—an ambulatory patient under observation beyond the confines of traditional points of care. An architecture was previously proposed to address the needs of a large and growing class of conditions and diseases amenable to tetherless care. The algorithms presented here, based on an O(1) Markov predictor, were implemented for this architecture and deployed on a set of Android devices. The work showed an ability to detect the rhythmic patterns in a given patient's connectivity resulting from his or her repetitive movements. This detection supported a prediction of future connectivity with greater than 80% accuracy and greater than 80% reliability. And this future knowledge provided an assurance for the system to introduce delay into the delivery of real-time data in order to achieve energy savings by exploiting the least costly means of connection.

8. REFERENCES

[1] Delay tolerant networking research group. http://www.dtnrg.org, 2013.

[2] A. Balasubramanian, R. Mahajan, and A. Venkataramani. Augmenting mobile 3g using wifi. In *MobiSys '10: Proceedings of the 8th international conference on Mobile systems, applications, and services*, pages 209–222, New York, NY, USA, 2010. ACM.

[3] F. Chang, J. Dean, S. Ghemawat, W. C. Hsieh, D. A. Wallach, M. Burrows, T. Chandra, A. Fikes, and R. E. Gruber. Bigtable: A distributed storage system for structured data. *ACM Trans. Comput. Syst.*, 26(2):4:1–4:26, June 2008.

[4] Y. Chon, E. Talipov, H. Shin, and H. Cha. Mobility prediction-based smartphone energy optimization for everyday location monitoring. In *Proceedings of the 9th ACM Conference on Embedded Networked Sensor Systems*, SenSys '11, pages 82–95, New York, NY, USA, 2011. ACM.

[5] P. Dillon and T. Znati. Towards an architecture for mobile healthcare. In *Local Computer Networks (LCN), 2012 IEEE 37th Conference on*, pages 260–263, 2012.

[6] P. Dillon and T. Znati. Analysis of messaging scheduling for information delivery in tetherless healthcare. http://cs.pitt.edu/techreports/?report=TR-2013-01, September 2013.

[7] E. Ertin, N. Stohs, S. Kumar, A. Raij, M. al'Absi, and S. Shah. Autosense: unobtrusively wearable sensor suite for inferring the onset, causality, and consequences of stress in the field. In *Proceedings of the 9th ACM Conference on Embedded Networked Sensor Systems*, SenSys '11, pages 274–287, New York, NY, USA, 2011. ACM.

[8] K. Fall and S. Farrell. Dtn: An architectural retrospective. *IEEE Journal on Selected Areas in Communications*, 26(5):828–836, June 2008.

[9] J. Huang, F. Qian, A. Gerber, Z. M. Mao, S. Sen, and O. Spatscheck. A close examination of performance and power characteristics of 4g lte networks. In *Proceedings of the 10th international conference on Mobile systems, applications, and services*, MobiSys '12, pages 225–238, New York, NY, USA, 2012. ACM.

[10] V. Jones, A. van Halteren, I. Widya, N. Dokovski, G. Koprinkov, R. Bults, D. Konstantas, and R. Herzog. Mobihealth: mobile health services based on body area networks. Technical Report TR-CTIT-06-37, Centre for Telematics and Information Technology, University of Twente, Enschede, January 2006. Imported from CTIT.

[11] G. Kambourakis, E. Klaoudatou, and S. Gritzalis. Securing medical sensor environments: The codeblue framework case. In *ARES '07: Proceedings of the The Second International Conference on Availability, Reliability and Security*, pages 637–643, Washington, DC, USA, 2007. IEEE Computer Society.

[12] M. Khabbaz, C. Assi, and W. Fawaz. Disruption-tolerant networking: A comprehensive survey on recent developments and persisting challenges. *Communications Surveys Tutorials, IEEE*, 14(2):607–640, 2012.

[13] J. Ko, J. H. Lim, Y. Chen, R. Musvaloiu-E, A. Terzis, G. M. Masson, T. Gao, W. Destler, L. Selavo, and R. P. Dutton. Medisn: Medical emergency detection in sensor networks. *ACM Trans. Embed. Comput. Syst.*, 10:11:1–11:29, August 2010.

[14] J. Ko, C. Lu, M. Srivastava, J. Stankovic, A. Terzis, and M. Welsh. Wireless sensor networks for healthcare. *Proceedings of the IEEE*, 98(11):1947 –1960, nov. 2010.

[15] K. V. Laerhoven, B. P. L. Lo, J. W. P. Ng, S. Thiemjarus, R. King, S. Kwan, H. werner Gellersen, M. Sloman, O. Wells, P. Needham, N. Peters, A. Darzi, C. Toumazou, and G. zhong Yang. Medical healthcare monitoring with wearable and implantable sensors. In *Proc. of the 3rd International Workshop on Ubiquitous Computing for Healthcare Applications*, 2004.

[16] D. Malan, T. Fulford-Jones, M. Welsh, and S. Moulton. Codeblue: An ad hoc sensor network infrastructure for emergency medical care. In *International Workshop on Wearable and Implantable Body Sensor Networks*, April 2004.

[17] A. Milenković, C. Otto, and E. Jovanov. Wireless sensor networks for personal health monitoring: Issues and an implementation. *Comput. Commun.*, 29(13-14):2521–2533, 2006.

[18] J. W. Ng, B. P. Lo, O. Wells, M. Sloman, C. Toumazou, N. Peters, A. Darzi, and G.-Z. Yang. Ubiquitous monitoring environment for wearable and implantable sensors (ubimon). In *Proceeding of The Sixth International Conference on Ubiquitous Computing*, September 2004.

[19] C. Otto, A. Milenkovic, C. Sanders, and E. Jovanov. System architecture of a wireless body area sensor network for ubiquitous health monitoring. *Journal of Mobile Multimedia*, 1(4):307–326, January 2006.

[20] V. Shnayder, B. rong Chen, K. Lorincz, T. Fulford-Jones, and M. Welsh. Sensor networks or medical care. Technical Report TR-08-05, Harvard University, 2005.

[21] A. Wood, J. Stankovic, G. Virone, L. Selavo, Z. He, Q. Cao, T. Doan, Y. Wu, L. Fang, and R. Stoleru. Context-aware wireless sensor networks for assisted living and residential monitoring. *Network, IEEE*, 22(4):26 –33, 2008.