Computer Communications 73 (2016) 47–55

Contents lists available at ScienceDirect

Computer Communications

journal homepage: www.elsevier.com/locate/comcom

# Efficient and Transparent Use of personal device storage in opportunistic data forwarding

Sayed Amir Hoseini[a,b,*], Azade Fotouhi[a,b], Mahbub Hassan[a,b], Chun Tung Chou[a], Mostafa H. Ammar[c,1]

[a] School of Computer Science and Engineering, University of New South Wales (UNSW), Sydney, Australia
[b] National ICT Australia
[c] Georgia Institute of Technology, Atlanta, GA, USA

ARTICLE INFO

*Article history:*
Available online 23 October 2015

*Keywords:*
Delay tolerant networks
Opportunistic networking
Bayesian decision
Mobile device storage

ABSTRACT

We consider a growing research trend of using personal mobile devices for forwarding opportunistic network data. Because personal device storage is meant to support user applications, opportunistic networks must use it in a manner that remains completely transparent to the user. One way to make a device's storage use transparent is to allow priority access to the storage to user applications, even if the storage is currently occupied by network data yet to be forwarded. This means that data given to a device waiting to be forwarded can be overwritten by application data and may, thus, be lost. In this paper we consider random access memory (RAM) as the primary storage location in a mobile device. We propose three algorithms of different sophistications to answer the question of how much data should be moved when a contact opportunity arises between two devices in such a way to first maximise the data transferred while minimising the probability that this data will be overwritten when applications claim priority access. We collect 33 h of high-resolution RAM usage traces of two real smartphones over a 3-day period under a variety of usage scenarios to evaluate and compare the performances of the proposed algorithms. Surprisingly, we find that autoregression forecasting of RAM usage cannot outperform the simplest algorithm that greedily occupies all of the RAM that is found unused at the time of contact. We show that Bayesian inference is very effective in minimising the risk of data loss in such uncertain environments and significantly outperforms the greedy approach as well as autoregression forecasting.



## 1. Introduction

Networks using opportunistic communication (also known as Delay or Disruption Tolerant Networks) are typically made up of mobile devices that are intermittently connected to each other [1]. Data in such networks is moved along from one node to another as nodes make contact. Receiving nodes that are not the data destinations will *store, carry, and forward* [2] the data until a contact is made with another node. A variety of routing protocols [3,4] have been proposed to determine the best sequence of contacts to use in such networks.

An important common feature of all these protocols is the requirement for intermediate nodes to store the data, potentially while the node is moving until it can be forwarded to another node. Surprisingly there is little in the research literature to discuss where this data needs to be stored and what the implications of using such storage are on the performance of opportunistic networks. We first note that there are primarily two types of intermediate nodes when opportunistic communication is used. Some nodes can be considered as infrastructure nodes that have been tasked with moving the data along. Examples include message ferries [5], data mules [6], and throwboxes [7]. In this case such nodes can be provisioned with memory that can be used exclusively to store data as it is carried and before it is forwarded.

In other systems, [8] user devices (such as smartphones and tablets) act as intermediate nodes. In such cases, the device's own memory can be used to store data until it can be forwarded. Random access memory (RAM), internal memory, and external memory (e.g., SD card) are possible memory options. Opportunistic networks require high throughput during contact opportunities in order to enable the transfer of significant amounts of data during even short contacts. Of the three types of memory available on a device, DRAM-based RAM technology provides the highest throughput [9–11]. Also

* Corresponding author at: School of Computer Science and Engineering, University of New South Wales (UNSW), Sydney, Australia. Tel.: +61(2) 93856198.
*E-mail addresses:* amirhoseini@cse.unsw.edu.au (S.A. Hoseini), azadef@cse.unsw.edu.au (A. Fotouhi), mahbub@cse.unsw.edu.au (M. Hassan), ctchou@cse.unsw.edu.au (C.T. Chou), ammar@cc.gatech.edu (M.H. Ammar).

[1] Mostafa Ammar's work is partially supported by National Science Foundation through grants NETS 1409589 and NETS 1161879.

http://dx.doi.org/10.1016/j.comcom.2015.10.003
0140-3664/

the cost of RAM has fallen drastically in recent years fuelling large RAM capacity in smartphones. While early smartphones could barely store more than a few kilo bytes in RAM, today's smartphones are shipped with at least 1 GB of RAM. Mobile social networking, content sharing, social discovery, pervasive and urban sensing and opportunistic computing are the most relevant applications which employ smart phone memory [12]. We will show in this paper that even when such applications are deployed, a mobile phone's RAM is typically underutilised, thus allowing this available capacity to be used as storage in support of opportunistic communication.

Unfortunately, a major issue arises from the fact that the smartphones are *personal devices*. As such, the phone storage is primarily dedicated to serve user applications. If operators wish to use this resource opportunistically, they must do so in a manner that remains *completely transparent to the user*. This raises a new challenge in opportunistic use of phone storage that has not been addressed before in the opportunistic communication literature.

One way to make a device's RAM use transparent is to allow applications priority access to the memory, even if the memory is currently occupied by data yet to be forwarded. This means that data given to a device waiting to be forwarded can be overwritten by application data and may, thus, be lost. In this paper we consider the question of how much data should be moved when a contact opportunity arises between two devices in such a way to first maximise the data transferred while minimising the probability that this data will be overwritten when applications claim priority access over a mobile device's memory.

To motivate the problem further, we collected and analysed a trace of available smartphone memory (RAM) from a user over three days (see Fig. 1(a)). As we can see, the unused memory can be very dynamic and vary significantly over a short span of time. For example, at about the 7900th sample, we observe 800 MB of unused RAM, but it drops to only 300 MB during the next few minutes. If a node came in contact with this phone at the 7900th and forwarded 800 MB of data, the phone would have to wipe out 500 MB of that data to make room for user applications. This highlights the risks involved in using smartphone memory for carrying opportunistic data.

The decision making can be considered as a dilemma between the utilisation of the unused memory and the reliability of data transport. If too much of the unused memory is utilised, there may be excessive data loss due to memory reclaim by the user. On the other hand, if low data loss is desired, unused memory may not be exploited efficiently. In this paper, we propose three decision making algorithms of different sophistication to answer the question of how much data should be moved when a contact opportunity arises between two devices in such a way to first maximise the data transferred while minimising the probability that this data will be overwritten when applications claim priority access. The first algorithm is the simplest algorithm that greedily occupies all of the RAM that it finds unused at the time of contact. The second is more sophisticated and uses autoregression forecasting to predict the minimum amount of memory that will still be available until the data is forwarded to the destination. In the third, we propose the use of Bayesian Decision Theory to minimise the risk of data loss using knowledge from past observations.

Using 33 h of high-resolution RAM usage traces of two real smartphones over a 3-day period under a variety of usage scenarios, we evaluate and compare the performance of the proposed algorithms. Surprisingly, we find that autoregression forecasting of RAM usage cannot outperform the simplest algorithm that greedily occupies all of the RAM that is found unused at the time of contact. We show that Bayesian inference is very effective in minimising the risk of data loss in such uncertain environments and significantly outperforms the greedy approach as well as autoregression forecasting.

The rest of the paper is organised as follows. The decision framework is defined in Section 2. We present the three decision algorithms in Section 3. Data collection and performance evaluation are presented in Section 4. Related work is discussed in Section 5. Section 6 concludes the paper.

## 2. Decision framework

In this section, we present a framework to define the decision that a node with data to be forwarded has to make when it comes in contact with a mobile phone. The aim of the forwarding node is to decide how much data to be offloaded to the mobile phone to carry. We assume that the smartphone receiving the data will carry it for a *transit* period before forwarding it along the next hop. We will consider the available memory as an integral multiple of a base chunk size $B$ megabytes. We use the following notations:

- $C$: memory capacity of the phone (it has a total $C \times B$ megabytes capacity)
- $k$: amount of memory not used by user applications (*available* memory) at time of contact; $0 \leq k \leq C$
- $j$: minimum amount of *available* memory during the entire transit period; $0 \leq j \leq C$
- $i$: amount of data the node decides to transfer to the phone; $0 \leq i \leq k$

The decision framework needs to consider the tradeoff between two types of losses: *data losses* ($L_D$) and *opportunity losses* ($L_O$). Data losses occur when the exploited unused memory to carry data is reclaimed by the user. In general, data losses have a higher chance of occurring if the source node offloads a higher amount of data to the phone. On the other hand, opportunity losses occur when there is available memory in the phone but the source does not make use of this unused memory. This tends to occur when too little data is offloaded to the phone. Note that although it is possible that new memory becomes available after the mobile phone has left the source location on the way to the destination location, we do not consider this as opportunity loss because there is no way that this memory could be used in the first place. Both losses are undesirable; the data loss decreases the reliability and retransmission of lost data is a waste of communication resource. On the other hand opportunity loss reduces the opportunistic network capacity. Given the tension between these two losses, the decision framework must find a way to balance between them.

Choosing $i$ is the decision that will influence the opportunity loss and the data loss. Given $i$, $j$, and $k$, we have the following three cases:

- $i = min(k, j)$: $L_O = 0; L_D = 0$
- $i < min(k, j)$: $L_O = min(k, j) - i; L_D = 0$
- $i > min(k, j)$: $L_O = 0; L_D = i - min(k, j)$

A good decision is one that minimises both losses, i.e., achieves good throughput (low opportunity loss) with minimum data loss. In the following section, we propose and discuss three different decision algorithms with contrasting features and properties.

## 3. Decision algorithms

In this section, we define three different decision algorithms, Greedy, Autoregression, and Bayesian. We also consider an Oracle algorithm to explain the best possible outcome that could be achieved with any decision making.

### 3.1. Oracle

This is the ideal decision with $i = j$. As such, there is no opportunity loss and no data loss. To make such decisions, the source node has to have the perfect knowledge of the minimum amount of memory that will remain available throughout the journey, i.e., it needs to know $j$ exactly. Clearly, this is not achievable, but it serves as a benchmark to compare different approaches.