Poster Abstract: Exploiting Multi-Channel Diversity to Speed Up Over-the-Air Programming of Wireless Sensor Networks

Weiyao Xiao and David Starobinski
Department of Electrical and Computer Engineering, Boston University
8 Saint Mary's St., Boston MA 02215, USA
{weiyao, staro}@bu.edu

Categories and Subject Descriptors
C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design – *Wireless Communication.*

General Terms
Algorithms, Performance, Design, Experimentation.

Keywords
Wireless, Reprogramming, Data Dissemination, Deluge, Motes.

1. INTRODUCTION
Wireless Sensor Networks consist of hundreds or thousands of ad-hoc tiny sensor nodes (motes) that are able to sense, compute, and communicate. Because of the large scale of sensor networks, programming them manually is a tedious and, sometimes, even impossible task. Thus, it is essential to have the capability to program them wirelessly.

Various schemes for over-the-air programming of sensor networks have been proposed, e.g., Deluge [1], MNP [2], and MOAP [3]. When node density increases, the performance of these schemes often degrades because of increasing traffic contention on the shared channel. While a number of mechanisms have been proposed to mitigate this problem, such as advertisement suppression [1] and selective sender selection [2], they do not solve it completely. In fact, the results of [1] as well as our experiments show that the completion time of Deluge, one of the most popular over-the-air programming schemes, considerably increases with node density.

In this paper, we propose to speed-up over-the-air programming by harnessing the multi-channel transceiving capability of motes. For instance, in the 902-928 MHz frequency region, there are as many as 25 non-overlapping channels (frequencies) over which motes can communicate. The idea, therefore, is to relieve network congestion by splitting the traffic among different channels.

The main challenge when trying to implement the above idea lies in the fact that standard motes are equipped with a single radio interface. This means that they can communicate only over one channel at a time. A static allocation of the channels is therefore infeasible, since nodes using one channel would lose connectivity with nodes using other channels.

Our main contribution in this paper is in the development of a general, dynamic method for exploiting multi-channel resources with single radio nodes. This method, described in more detail in Section 2, divide nodes into different *groups*. Nodes from all the groups can always communicate over a *default* channel. However, whenever possible, nodes belonging to the same group temporarily switch and communicate between themselves over an *auxiliary* channel that is unique to each group. We have devised a new over-the-air programming scheme for TinyOS motes using this method, called *Multi-Channel Deluge*. We have implemented Multi-Channel Deluge on a testbed of MICA2 motes operating in the 902-928 MHz frequency region. Our initial experimental results show that Multi-Channel Deluge can reduce the programming time by as much as 60% compared to the standard implementation of Deluge.

2. MULTI-CHANNEL DATA DISSEMINATION USING SINGLE RADIOS
In this section, we present our method for network programming over multiple channels using single radios. This method is general and can be implemented, at least in principle, in conjunction with any existing over-the-air programming scheme. We assume that the bandwidth used for network programming consists of K non-overlapping channels, indexed from 1 to K. We accordingly divide the network into K disjoint groups of nodes, indexed from 1 to K. Without loss of generality, we define channel 1 to be the default channel and group 1 to be the default group. We require that group 1 forms a connected dominating set, that is, all the nodes in the network can always directly communicate with at least one node belonging to group 1. We also assume that at least one node from group 1 (e.g., the base station) has the new version of the program. The group membership of nodes not belonging to the connected dominating set is obtained through a simple hashing operation on the node ID (e.g., for the case of 2 channels, nodes with even IDs belong to group 1 and nodes with odd IDs belong to group 2). This procedure could certainly be optimized, for instance, by taking into account topological properties of the network. However, even without such optimization, significant performance improvement over single channel network programming can be achieved as shown by our experimental results.

Initially, all the nodes advertise the current version of their program on channel 1. Nodes check whether any advertisement they receive comes from a node belonging the same group (this can be done by looking at the node ID of the sender). Nodes belonging to the default group operate the same way as in existing over-the-air programming schemes. If a node from a non-default group i (i.e., i=2,3,…,K) receives an advertisement for a newer program (or a page) from a node belonging to the same group i, then it sends a request to that node. After getting one or more requests from nodes belonging to the same group i, the node that sent the original advertisements sends a second message announcing that it will switch to auxiliary channel i. At that point, all the nodes from group i interested in receiving the data will switch to channel i as well. A node will stay in channel i, as long as it either sends or receives new data from other nodes in the same channel. Otherwise, after some time-out τ, it will switch back to the default channel. Finally, if a node from non-default group i

SenSys'05, November 2–4, 2005, San Diego, California, USA.
ACM 1-59593-054-X/05/0011.

does not receive an advertisement for newer data from a node belonging to the same group, then it will attempt to receive this data from a node belonging to the default group. Since the default group forms a connected dominating set, one is guaranteed that the new program will ultimately reach all the nodes.

3. MULTI-CHANNEL DELUGE

We have implemented the multi-channel programming method described above into the Deluge framework [2]. Some of the major modifications made to Deluge are as follows:

1. Advertisement suppression mechanism: A node suppresses its advertisement only if the same advertisement was sent earlier by a node within the same group. We note that this mechanism may increase the number of advertisements sent on the default channel, compared to Deluge. However, remember that nodes belonging to non-default groups typically communicate over an auxiliary channel and send advertisements over that channel.

2. Requests: Nodes prefer sending requests to nodes belonging to the same group. However, if no node from the same group advertises a new program/page while a node from the default group does send a new advertisement, then a request is sent to that latter node and data is transferred over the default channel.

3. Channel switching: If a node receives one or more requests from nodes in the same non-default group, then it sends an announcement message that it will switch channel. Nodes from the same group interested in receiving the data switch to that channel as well.

4. Operation on auxiliary channel: Once nodes switch to an auxiliary channel, they perform the same operations as in Deluge (i.e., send advertisements and requests, send and receive new data, etc.). However, if a node does not receive or send any new portions of a program for a period of time exceeding some threshold τ, then it switches back to the default channel.

4. EXPERIMENTAL RESULTS

In this section, we present preliminary experimental results comparing the performance of Deluge and Multi-Channel Deluge. Our testbed consists of MICA2 motes operating in the 902-928 MHz frequency region. The network diameter is one, i.e. all the motes can directly communicate one with another. We conjecture that the performance improvement achieved with Multi-Channel Deluge would be similar or even higher in multi-hop networks, where the hidden node problem can detrimentally impact the performance of shared-medium access protocols [1]. We use the default parameters of Deluge and, for the case of Multi-Channel Deluge, set the parameter τ to 7 seconds. In each of our experiments, we disseminate a new program of 22 pages with 24288 bytes of data. Each point shown in the graphs represents an average taken over three identical experiments.

1. Single-source experiments: In our first suite of experiments, we consider the case where, initially, only one node has the new version of the program. We assume that Multi-Channel Deluge uses K=2 channels.

Figure 1 shows the completion times of Deluge and Multi-Channel, as a function of the total number of nodes in the network. We observe that both schemes perform comparably when the network density is low (i.e., up to 10 nodes). However, at higher node density, Multi-Channel Deluge is much faster. For instance, for a 25-node network, Multi-Channel Deluge completes in slightly more than 250 seconds, while Deluge takes close to 400 seconds.

The main reason why Deluge slows down at higher node density is that, due to increasing traffic contention and interferences, the sender must spend significant time on packet retransmission. With Multi-Channel Deluge, once a node from a non-default group completes receiving a new page, it will start transmitting it on an auxiliary channel, if other nodes from the same group need it. This procedure relieves congestion on the default channel and substantially speeds-up the entire data delivery process.

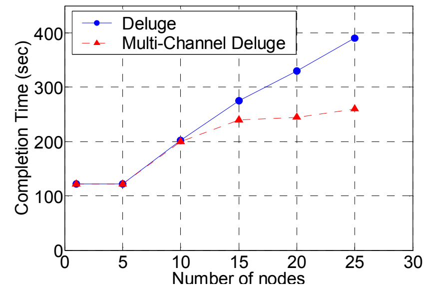

Fig. 1. Completion time with a single source

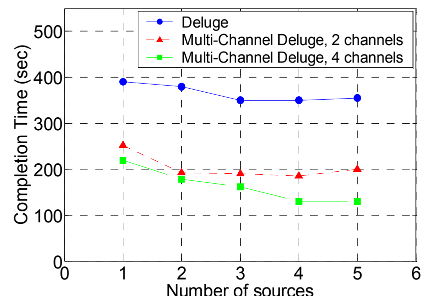

Fig. 2. Completion time with multiple sources

2. Multi-source experiments: We consider now the case where multiple nodes initially have the new version of the program. The number of nodes in the network is kept fixed to 25 motes. We evaluate the performance of Deluge and that of Multi-Channel Deluge with K=2 and K=4 channels. For the case of Multi-Channel Deluge, each source belongs to a different group (if the number of sources is greater than the number of groups, then at least one source belongs to each group).

Figure 2 shows that while Deluge benefits only marginally from the presence of multiples sources, Multi-Channel Deluge effectively utilizes them and achieves significant reduction in the network programming time. Comparing Figures 1 and 2, we observe that the completion time of Multi-Channel Deluge with N nodes and K channels and sources is approximately equivalent to that of Deluge with N/K nodes. These results illustrate the great potential of our method in improving the performance and scalability of over-the-air programming mechanisms at high network density.

5. ACKNOWLEDGMENT

This work is supported in part by the US National Science Foundation under grants ANI-0132801 and CNS-0435312.

6. REFERENCES

[1] J. W. Hui and D. Culler. The Dynamic Behavior of a Data Dissemination Protocol for Network Programming at Scale. *Proceedings of ACM SenSys '04*, Baltimore, MD.

[2] S. Kulkarni and L. Wang. MNP: Multihop Network Reprogramming Service for Sensor Networks. *Proceedings of ICDCS '05*, Columbus, OH.

[3] T. Stathopoulos, J. Heidemann, and D. Estrin. A Remote Update Code Mechanism for Wireless Sensor Networks, UCLA Tech Report CENS-TR-30, 2003.