ARTICLE IN PRESS
JID: ADHOC [m5G;October 3, 2016;11:4]
Ad Hoc Networks 000 (2016) 1–11
Contents lists available at ScienceDirect
Ad Hoc Networks
journal homepage: www.elsevier.com/locate/adhoc

CodeDrip: Improving data dissemination for wireless sensor networks with network coding

Nildo dos Santos Ribeiro Júnior[a], Rodrigo C. Tavares[a], Marcos A.M. Vieira[a,*], Luiz F.M. Vieira[a], Omprakash Gnawali[b]

[a] *Universidade Federal de Minas Gerais Belo Horizonte, Brazil*
[b] *Department of Computer Science, University of Houston, Houston, TX, USA*

ARTICLE INFO

Article history:
Received 12 August 2015
Revised 12 September 2016
Accepted 28 September 2016
Available online xxx

Keywords:
Wireless sensor networks
Network coding
Dissemination protocol
Experimental and prototype results

ABSTRACT

Network Coding is a technique that combines packets in the network with the potential to mitigate packet lost. In wireless communication systems, traditionally, dropped packets are recovered using retransmissions. By combining packets using network coding, it is possible to recover the transmitted information without needing to retransmit all the lost packets to all the nodes. In this paper, we present a show-case that applying Network Coding to data dissemination for Wireless Sensor Networks provides benefits even for small values. CodeDrip is a data dissemination protocol with Network Coding capability. Dissemination is typically used to query nodes, send commands, and reconfigure the network. CodeDrip utilizes Network Coding to improve energy efficiency, reliability, and speed of dissemination. While previous work in combining network coding and dissemination focused on bulk data dissemination, we optimize the design of CodeDrip for dissemination of small values. We perform extensive evaluation of CodeDrip on simulations and a large-scale testbed and compare against the implementations of Drip, DIP and DHV protocols. We evaluate with more than 350 physical sensor nodes. Results show that CodeDrip is faster, smaller and sends fewer messages than Drip, DHV and DIP protocols.

© 2016 Elsevier B.V. All rights reserved.

1. Introduction

Network Coding [3] is a technique that combines packets in the network with the potential to increase throughput, decrease energy consumption, and reduce the number of messages in Wireless Communication Systems.

Wireless Sensor Network (WSN) consists of a large number of nodes with sensing, computation, and wireless communication capability. This sensor network is typically deployed to collect data from the environment or other physical spaces. Wireless communication and energy efficiency are key requirements for WSNs, especially in applications where we retrofit existing infrastructure.

Many WSN applications require the capability to send messages from a base station or controller node to all the nodes in the network. This type of communication pattern is called *dissemination*, or one-to-many. Dissemination is typically used to send commands, query, reconfigure and reprogram the network. A data dissemination protocol for sensor networks needs to overcome several challenges. First, the energy in each sensor node is limited by the battery or energy harvesting capacity, thus it is important to save energy to increase the sensor node's operational lifetime. Second, sensor nodes typically do not have powerful CPUs, so they might not be capable of executing complex matrix operations or complex communication protocols. Finally, wireless communication is susceptible to transmission errors and packet loss. A dissemination protocol should not only be reliable and energy efficient, but also fast.

In this paper, we present CodeDrip,[1] a data dissemination protocol for Wireless Sensor Networks. CodeDrip uses Network Coding to improve energy efficiency, reliability, and speed of dissemination. Instead of simply retransmitting received data packets, sensor nodes combine various packets into one, and retransmit the combined packet to its neighbors. Therefore, packet loss is mitigated since lost packets might be recovered through the decoding of others combined packets. By avoiding retransmission, the dissemination process might finish faster.

Existing data dissemination protocols for Wireless Sensor Networks present a tradeoff: save energy at the expense of dissemina-

* Corresponding author.
E-mail addresses: nildo@dcc.ufmg.br (N.d.S. Ribeiro Júnior), rodrigoct@dcc.ufmg.br (R.C. Tavares), mmvieira@dcc.ufmg.br (M.A.M. Vieira), lfvieira@dcc.ufmg.br (L.F.M. Vieira), gnawali@cs.uh.edu (O. Gnawali).

[1] This work is based on preliminary conference version [8].

http://dx.doi.org/10.1016/j.adhoc.2016.09.023
1570-8705/© 2016 Elsevier B.V. All rights reserved.

Please cite this article as: N.d.S. Ribeiro Júnior et al., CodeDrip: Improving data dissemination for wireless sensor networks with network coding, Ad Hoc Networks (2016), http://dx.doi.org/10.1016/j.adhoc.2016.09.023

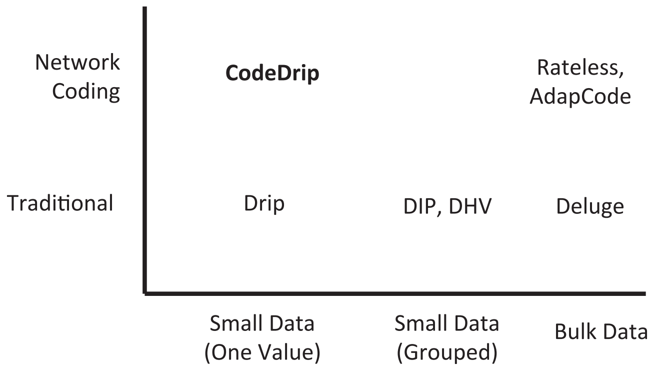

Fig. 1. Selected classes of dissemination protocols in sensor network. CodeDrip uses network coding to make dissemination of small data efficient and fast.

tion speed. These protocols use transmission of summaries or version numbers and attempt to selectively transmit the missing data to avoid redundant transmission. While this strategy saves energy, it could incur large delays. Through extensive experiments, we find that CodeDrip provides faster data dissemination while transmitting fewer messages than most previous approaches.

Network Coding is not a new idea in wireless communication. However, previous work such as COPE [16], cannot be applied to WSNs because these algorithms require large memory overhead. Network coding schemes such as Rateless Deluge [13] and AdapCode [14] have been previously proposed in wireless sensor networks and are shown to have low memory and computational overhead. However, these WSN dissemination protocols are designed for bulk data dissemination. There has been no previous study of effectiveness of network coding in dissemination of *small values*. Previously, it was thought that there would be limited opportunity to combine packets in dissemination of small values, hence the focus on bulk data dissemination. We identify the opportunity to make dissemination of small values efficient with network coding and fill this gap in sensor network protocol design space.

Our main contributions are as follows. First, we present the design and implementation of CodeDrip. Second, we study the performance of CodeDrip through extensive simulation and testbed experiments on the KanseiGenie testbed [26]. We evaluate CodeDrip with more than 350 physical sensor nodes. Our work is a realistic show-case that Network Coding benefits real-world applications. Third, we compare CodeDrip to the data dissemination of small values with Drip, DIP [23] and DHV [5] and quantify the Network Coding gain and show that Network Coding is useful even for dissemination of small values.

Our work is organized as follows. In the next section, we present the related work. CodeDrip is explained in Section 3. We present the simulation results in Section 4 and the testbed experiments in Section 5. We conclude in Section 6.

2. Related work

There is a large body of work in dissemination protocols for WSN. Fig. 1 summarizes the major classes of dissemination protocols. In this section, we describe how some of the dissemination protocols fit in the design space of dissemination protocols and application of network coding in dissemination.

2.1. Dissemination of small values

Trickle [21] is used as a building block for a number of dissemination protocols that propagate code or small values in WSN. Trickle has two key features that allows it to be efficient: the timer control algorithm and duplicate suppression. Trickle timer doubles its interval every time it fires. Thus, over the long run, the interval converges to a very large maximum value. The timer can be reset to a small value when a new message needs to be sent. Trickle uses version numbers to detect and suppress duplicate transmissions. A node periodically broadcasts its version but stays silent and increases the interval if it hears several messages from its neighbors containing the same version number. When a node receives a new version number, the node resets the timer and transmits the message. CodeDrip uses Trickle timer in its design.

Fig. 2. Drip example. There are three values to be disseminated. Each value is independently associated with a Trickle timer. Each packet has a value (rectangles on the left side). Each dot represents when the timer fires and a message is sent.

Some works [9,27] study the Trickle performance. In [27], the authors present an analytical study of the performance of Trickle algorithm for data dissemination. In [9], the authors analyzed Trickle parameters and proposed a modification when a new node joins the network. Moreover, the CTP work[11] evaluates the Trickle adaptive beaconing and presents some discussion about Trickle timers and different parameters.

Drip [28] is the simplest data dissemination protocol that uses Trickle timer. Each time an application transmits a message with Drip, a new version number is used. The new version number causes the protocol to reset the Trickle timer and thereby transmissions in the network to disseminate the new value. Redundant transmissions are detected using version numbers and suppressed. When the application does not inject new messages, the timer interval increases which causes the control overhead to level off. When a new message is injected, the new version number causes the timer to reset and the nodes disseminate the message. Fig. 2 shows how Drip works when it is used to disseminate three values. Dissemination of each message is paced by its own Trickle timer.

DIssemination Protocol (DIP) [23] is a data discovery and dissemination protocol. DIP continuously measures network conditions and estimates whether each data item requires an update. In DIP, a node periodically broadcasts a summary message, containing hashes of its keys and versions. A hash-tree based algorithm detects if there is an update. DIP scales logarithmically with the total number of items. DHV [5] is a code consistency maintenance protocol. DHV's key contribution is its technique to efficiently determine when to perform code updates. DHV detects and identifies which code item need updates at the bit level. DHV uses the Trickle timer to control transmission rate and duplicate suppression.

DIP and DHV are examples of dissemination protocols that operate at the level of a group of messages (for example, to compute summaries). On the other hand, Drip operates at the granularity of a single message. In DIP and DHV, all nodes must agree on a fixed set of data item identifiers before dissemination. DIP and DHV can scale to a large number of data item updates, however perform worse than Drip on small number of data items or updates [12].

2.2. Bulk and middle-sized data dissemination

A different set of dissemination protocols have been proposed for middle or large-sized objects. Maté [19] and Tenet [24] optimize the design of their dissemination protocols for middle-sized objects. Maté virtual machine disseminates code capsules to install

Download English Version:

https://daneshyari.com/en/article/4953722

Download Persian Version:

https://daneshyari.com/article/4953722

Daneshyari.com